ВИДАВНИЦТВО
РАНОК

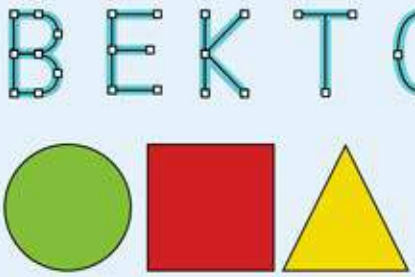
Інтернет-
підтримка

6

ІНФОРМАТИКА

КОМП'ЮТЕРНА

Векторна графіка



- \+ Невеликий обсяг даних
- \+ Збереження якості при масштабуванні
- \+ Легкість редагування та модифікації зображень
- — Схематичність зображення
- — Неприродність кольорів

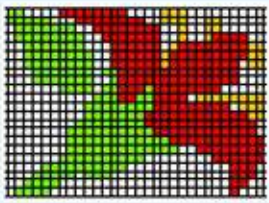
Векторні зображення

- Складаються з об'єктів, описаних математично
- Менші обсяги файлів. Обсяг залежить від кількості об'єктів на зображенні
- Можна збільшувати без погіршення якості
- Не дають змогу точно передати перехід від одного кольору до іншого
- Застосовують для зберігання креслень, графіки, рисунків із чіткими контурами

ГРАФІКА

Растрова графіка

РАСТР



- \+ Реалістичність зображень
- \+ Природність кольорів
- \+ Можливість отримання зображень за допомогою спеціальних пристроїв
- — Великий обсяг даних
- — Пікселізація зображення при збільшенні масштабу
- — Складність редагування окремих елементів зображення

Растрові зображення

- Складаються з масивів пікселів
- Більші обсяги файлів. Обсяг залежить від розміру зображення
- У разі збільшення зображення якість погіршується
- Дають змогу отримати зображення фотографічної якості
- Застосовуються для зберігання фотографій, творів живопису тощо

6

ІНФОРМАТИКА

Підручник для 6 класу
закладів загальної середньої освіти

Рекомендовано
Міністерством освіти і науки України

Харків
Видавництво «Ранок»
2019

УДК 004:37.016(075.3)
І-74

Авторський колектив:
Олена Бондаренко, Василь Ластовецький,
Олександр Пилипчук, Євген Шестопалов

Рекомендовано Міністерством освіти і науки України
(наказ Міністерства освіти і науки України від 12.04.2019 № 472)

Видано за рахунок державних коштів. Продаж заборонено

І-74 **Інформатика** : підруч. для 6 кл. закл. загал. серед. освіти / [О. О. Бондаренко, В. В. Ластовецький, О. П. Пилипчук, Є. А. Шестопалов]. — Харків : Вид-во «Ранок», 2019. — 160 с. : іл.

ISBN 978-617-09-5188-5

УДК 004:37.016(075.3)

Інтернет-підтримка
Електронні матеріали
до підручника розміщено на сайті
interactive.ranok.com.ua

ISBN 978-617-09-5188-5

ДОРОГІ ШЕСТИКЛАСНИКИ ТА ШЕСТИКЛАСНИЦІ!

Ви тримаєте в руках підручник з інформатики, призначений саме для вас, учнів і учениць 6 класу. У 5 класі ви ознайомилися з основними поняттями інформатики, навчилися працювати з комп'ютером, редагувати і форматувати текстові документи. Ви дізналися про особливості будови і роботи комп'ютерних мереж, приступили до вивчення основ мови програмування Python, опанували складання та виконання алгоритмів із розгалуженнями і повтореннями.

У цьому навчальному році на уроках інформатики вас очікує чимало цікавого та корисного, а пропонований підручник буде вашим надійним помічником. Як же з ним працювати?

Підручник складається з трьох розділів. На початку кожного розділу ви знайдете рубрику *«Повторюємо»*. Вона допоможе вам згадати відомості, які ви вивчали в попередніх класах і які будуть корисні для засвоєння нового матеріалу.

Розділ 1 «*Комп'ютерна графіка*» присвячений знайомству з особливостями побудови й опрацювання растрових і векторних зображень, розділ 2 «*Комп'ютерні презентації*» присвячений створенню презентацій з елементами управління показом, розробці сценарію презентацій та добиранню стильового оформлення слайдів, а розділ 3 «*Алгоритми та програми*» — ознайомленню з принципами об'єктно-орієнтованого програмування, створенню ігрових програм та ін.

Розділ підручника складається з параграфів і практичних робіт за темами розділу. Кожен параграф містить теоретичні відомості за темою уроку, приклади практичного застосування отриманих знань, питання для самоперевірки, вправи.

Питання для самоперевірки допоможуть з'ясувати, чи зрозуміли ви вивчений матеріал, а також підготуватися до виконання вправ і практичних робіт за комп'ютером.

Вправи складаються із завдань теоретичного і практичного спрямування. Виконуючи їх, ви навчитеся краще працювати за комп'ютером.

Оцінити свої знання, вміння та навички вам допоможе *комп'ютерне тестування* з автоматичною перевіркою результату. Його можна пройти на сайті interactive.ranok.com.ua

У підручнику ви знайдете практичні роботи з покроковим описом. Щоб виконати кожну з них, потрібно повторити матеріал, вивчений на попередніх уроках, — тоді ви зможете успішно застосувати свої знання, виконуючи завдання за комп'ютером.

Бажаємо натхнення та успіхів!

У тексті підручника використано такі позначення:


Запам'ятайте


Розгляньте приклад


Ознайомтеся з цікавою інформацією

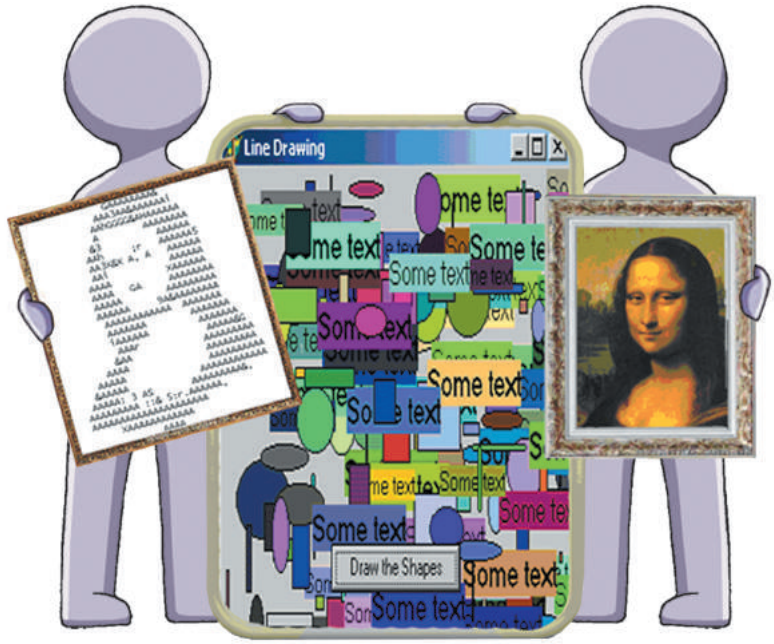

Знайдіть відповідь в Інтернеті


Виконайте практичне завдання за комп'ютером


Виконайте завдання підвищеної складності


Виконайте завдання в парах

РОЗДІЛ 1
КОМП'ЮТЕРНА ГРАФІКА



ПОВТОРЮЄМО

Сьогодні зображення, опрацьовані за допомогою комп'ютера (схеми, малюнки, фотографії тощо), ми бачимо скрізь: у книжках і журналах, на веб-сторінках сайтів, у комп'ютерних іграх тощо. Зазвичай такі зображення називають *комп'ютерною графікою*.

Працюють із зображеннями за допомогою спеціальних програм — *графічних редакторів*. Ви вже познайомилися з деякими з них, умієте створювати прості зображення та додавати до них написи; знаєте, як зберігати готовий малюнок на диску.

Для введення зображення в комп'ютер є різні пристрої: *сканер*, *цифровий фотоапарат*. Надрукувати малюнок на папері можна за допомогою *принтера*.

1. Що називають комп'ютерною графікою?
2. Що називають графічним редактором?
3. З яким графічним редактором ви працювали?
4. Опишіть інструменти графічного редактора.
5. Як ввести зображення в комп'ютер?
6. Як отримати паперову копію комп'ютерного малюнка?

У цьому розділі ви дізнаєтеся про основні поняття комп'ютерної графіки, з'ясуєте особливості побудови й опрацювання растрових і векторних зображень, навчитеся використовувати шари для створення зображень.

§ 1. Основні поняття комп'ютерної графіки. Растрова графіка

Комп'ютерна графіка завдяки розвитку інформаційних технологій набуває дедалі більшого поширення в усьому світі. Сьогодні її застосовують у різних сферах людської діяльності: у науці, медицині, рекламі, поліграфії, на виробництві та ін. Добре відомі здобутки комп'ютерної графіки у створенні спецефектів у кінофільмах, комп'ютерних іграх тощо.

Комп'ютерна графіка — це розділ інформатики, у якому вивчаються методи створення й опрацювання зображень за допомогою комп'ютера.

Уся інформація, опрацьовувана комп'ютером, кодується за допомогою чисел. Але як закодувати числами малюнок? Для цього є два способи: *растровий* і *векторний*.

Растрові зображення

Намалюйте на аркуші в клітинку квітку, зафарбовуючи певним кольором лише цілі клітинки. Ви отримаєте зображення, схоже на те, що наведено на рис. 1.1. Подібну структуру має зображення на екрані монітора: воно складається з окремих крапок, які називають **пікселями** (від англ. *Picture Element* — елемент малюнка).

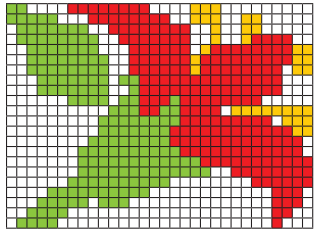


Рис. 1.1

Пікселі малі, тому на екрані ми бачимо суцільне зображення.

Зображення, подане як набір пікселів, називають **растровим**, або **точковим**. Піксель є найменшим елементом растрового зображення.

Кожен піксель може набувати тільки одного кольору. Змінити колір частини пікселя неможливо.

Кодування растрових зображень

Растрове зображення нескладно закодувати числами.

Позначимо кожен колір числом і отримаємо кодову таблицю:

Колір	Код	Колір	Код	Колір	Код	Колір	Код
Чорний	0	Зелений	2	Червоний	4	Жовтий	6
Синій	1	Коричневий	3	Фіолетовий	5	Білий	7

Тоді кодом растрового зображення буде послідовність чисел: перші два числа — кількість пікселів по довжині і ширині малюнка, наступні — коди кольорів пікселів, перелічені зліва направо рядок за рядком:

30, 22, 2, 2, 7, 7, 7, 7, 4, 4, 4, 4, 4, 4, 4, 4, 7, 7, 7, 7, 6, 6, 6 і т. д.

Растровий спосіб кодування використовується для опрацювання фотографій, малюнків, сканованих зображень тощо.

Для роботи з растровою графікою є багато програм. Вони відрізняються набором засобів для опрацювання зображень: для початківців — Paint, KolourPaint; для професіоналів — Adobe Photoshop, GIMP та ін.


Paint


KolourPaint

Adobe Photoshop


GIMP

Рис. 1.2

Растрова графіка має як переваги, так і недоліки. *Основною перевагою* є можливість працювати з фотографічними та сканованими зображеннями, *недоліком* — погіршення якості зображення в разі змінення його розмірів або обертання на певний кут.

Властивості растрових зображень

Розміри растрового зображення визначає кількість пікселів по горизонталі й вертикалі. Від кількості пікселів залежить якість зображення, адже чим пікселів більше, тим більше окремих деталей за їх допомогою можна зобразити.

Типові розміри екрана монітора в пікселях такі: 800 × 600, 1024 × 768, 1240 × 1024 і більше. Цифрове фото може мати розміри 2048 × 1536, 4320 × 3240 пікселів тощо.

Надруковане зображення повинно бути якісним, а отже, пікселі — якомога меншими за розмірами.

Величину, що показує, скільки пікселів розміщується на одиниці довжини зображення, називають його **роздільною здатністю**, або **роздільністю**, і найчастіше вимірюють у «точках на дюйм» (англ. *dpi* — *dots per inch*).

Окрім розміру, зображення можна охарактеризувати кількістю можливих кольорів. Чим більше кольорів використано в зображенні, тим краще воно може відтворювати вигляд реальних об'єктів.

Слід пам'ятати, що для кодування більшої кількості кольорів будуть потрібні довші коди, що спричиняє збільшення обсягу файла.

Таким чином, під час створення або збереження растрового зображення слід вказувати його розміри та колір кожного пікселя. Для точнішого відтворення слід задавати також роздільність.

Растровий графічний редактор GIMP

Нескладні малюнки зручно створювати за допомогою простого графічного редактора, наприклад Paint. Проте є програми, наприклад GIMP, що мають більше можливостей і дозволяють опрацьовувати значно складніші зображення.

Графічний редактор GIMP розробляється інтернет-спільнотою, вільно розповсюджується і має різноманітні засоби для опрацювання растрових зображень. Програма є багатоплатформною, тобто одночасно виходять версії для різних операційних систем (Windows, Linux та ін.).

GIMP дозволяє створювати й опрацьовувати растрові малюнки, покращувати якість фотографій та сканованих зображень, розробляти колажі, тобто об'єднувати кілька зображень в одне

(наприклад, додавати рамку до фотографії) тощо. Завдяки цьому GIMP використовують різні фахівці в галузі комп'ютерної графіки (фотографи, дизайнери та ін.).

Для того щоб запустити графічний редактор GIMP, потрібно двічі клацнути його піктограму. Після цього з'явиться головне вікно програми (рис. 1.3).

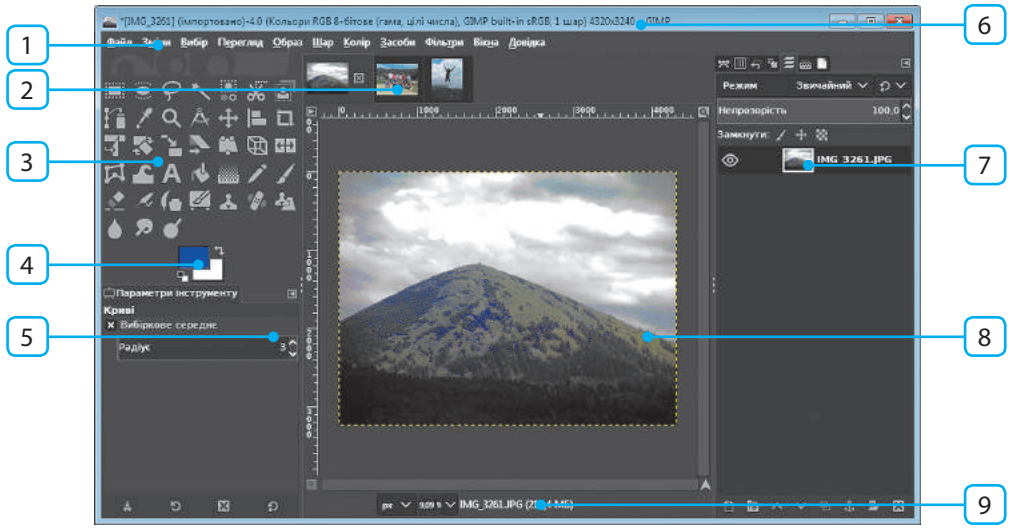


Рис. 1.3

Розглянемо основні елементи головного вікна програми:

1 — рядок меню
2 — вкладки відкритих зображень
3 — панель інструментів
4 — вибір кольорів
5 — параметри інструмента
6 — рядок заголовка
7 — діалогові вікна
8 — робоче поле
9 — рядок стану

Якщо після запуску програми з'явиться кілька окремих вікон, то їх можна об'єднати в одне командою меню Вікна → Одновіконний режим.

! Вигляд вікна програми може відрізнятися від наведеного на рис. 1.3, оскільки GIMP дозволяє змінювати розташування й приховувати більшість елементів керування.

Створення й опрацювання зображення

Для **створення нового зображення** треба вибрати команду Файл → → Створити, зазначити в діалоговому вікні (рис. 1.4) ширину й висоту майбутнього зображення в пікселях.

Можна також зі списку Шаблони вибрати один із готових варіантів розмірів зображення.

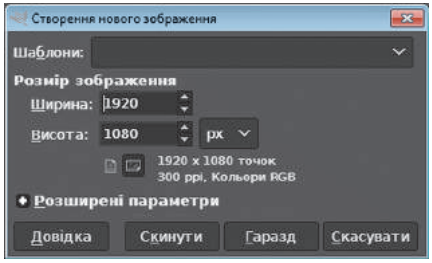


Рис. 1.4

Вибір кольорів

Для роботи із зображенням можна вибрати два кольори: *колір переднього плану* та *колір тла*. Унизу панелі інструментів містяться засоби для керування цими кольорами (рис. 1.5).

1 — зразок кольору переднього плану
2 — відновлення початкових кольорів
3 — обмін кольорів
4 — зразок кольору тла

Рис. 1.5

Якщо клацнути один зі зразків кольорів, то відкриється діалогове вікно для вибору відповідного кольору (на рис. 1.6 — фрагмент вікна).

У діалоговому вікні зручно **вибирати колір** у режимі колірного колеса, що вмикається кнопкою  . Для цього слід клацнути на кільці потрібний тон кольору, а потім клацнути в трикутнику потрібний відтінок.

Рис. 1.6

Огляд засобів керування

На панелі інструментів містяться кнопки різноманітних інструментів. Щоб **вибрати потрібний інструмент**, слід клацнути відповідну кнопку. У вікні параметрів, яке розміщено під панеллю інструментів, можна переглядати і змінювати властивості активного інструмента.

Доступ до більшості засобів для роботи із зображенням здійснюється через діалогові вікна (позначка 7 на рис. 1.4). Їх розта-

шування можна змінювати. Якщо потрібного діалогового вікна на екрані немає, його слід увімкнути, вибравши назву з меню Вікна → Діалоги з підтримкою прикріплення.

Для більш зручного перегляду зображення, відшукування дрібних дефектів іноді доцільно збільшити чи зменшити масштаб, натиснувши клавішу Ctrl і прокрутивши коліщатко миші.

> ! Зміна масштабу впливає не на саме зображення, а лише на його вигляд на екрані.

Перейти до потрібної частини малюнка можна за допомогою смуг прокручування, розташованих унизу та праворуч від робочого поля із зображенням.

Інструменти малювання

Інструменти GIMP за призначенням можна поділити на інструменти малювання, виділяння, заповнення, перетворення та ін.

Розглянемо детальніше інструменти малювання.

Інструмент Пензель дозволяє малювати за допомогою «пензлів» різної форми. На рис. 1.7 показано частину панелі параметрів інструмента Пензель.

Зі списку Режим (1) можна вибрати, як саме пензель впливатиме на зображення. Кнопкою (3) розкривається список пензлів, у полі (4) зазначається назва вибраного пензля. Регулятори призначено для змінення щільності (непрозорості) (2), розміру (5) і співвідношення сторін (6) пензля.

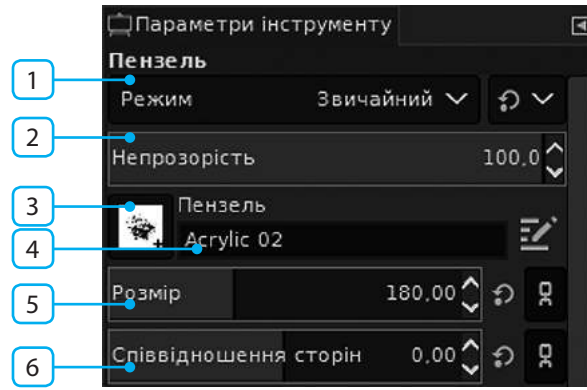


Рис. 1.7

2 Є багато режимів малювання. У режимі Звичайний «мазки» пензля просто накладаються на малюнок, у режимі Розсіяне світло пензель ніби освітлює малюнок ліхтариком вибраного кольору.

Інструменти Пензель, Олівець, Аерограф і Гумка використовують спільний набір «пензлів»:

Значок	Інструмент	Опис
	Олівець	Відрізняється тим, що накреслені ним лінії не мають напівпрозорих ділянок
	Аерограф	Діє аналогічно **Пензлю**, але щільність його сліду залежить від тривалості утримання кнопки миші
	Гумка	Дозволяє стирати ділянки малюнка. Якщо для шару увімкнено прозорість, то слід **Гумки** стане прозорим (див. далі), інакше він буде зафарбований кольором тла

Кадрування зображення

Графічний редактор GIMP дає змогу кадрувати зображення, тобто дуже точно обрізати його інструментом Кадрування  .

Щоб **вилучити зайві ділянки малюнка**, слід:

1) вибрати інструмент Кадрування;
2) у робочому полі перетягнути прямокутник на ту частину зображення, яку потрібно залишити;
3) уточнити розміри та положення прямокутника, перетягуючи бічні смуги або весь прямокутник;
4) клацнути всередині прямокутника.

Збереження зображення

Якщо малюнок не завершено, його потрібно **зберегти для подальшої роботи.** Для цього слід:

1) вибрати команду меню Файл → Зберегти;
2) у діалоговому вікні Збереження зображення, що відкриється, вибрати папку і назву файла;
3) клацнути кнопку Зберегти.

Файл буде збережено у форматі XCF — спеціальному форматі графічного редактора GIMP.

Якщо роботу над малюнком завершено, то його слід **експортувати** в потрібний растровий формат. Для цього потрібно вибрати команду меню Файл → Експортувати, а потім у діалоговому вікні — папку, назву файла і формат для збереження.

Формати файлів растрових зображень

Згадаємо, що формат файла — це набір правил, за якими дані записуються у файл. Існує багато форматів файлів для запису растрових зображень, що відрізняються розміром файла, способом розміщення даних у файлі, якістю зображення та ін.

Розглянемо деякі найпоширеніші растрові формати.

- Формат PNG дозволяє зберігати растрові зображення зі стисканням без втрати якості. Залежно від типу він підтримує від 65 536 до понад 4 млрд кольорів, а також прозорість. Формат користується популярністю в Інтернеті.
- Формат JPG використовується для стиснення графічних даних за рахунок втрати якості зображення: менший файл — нижча якість. Завдяки порівняно невеликим розмірам файлів формат часто застосовується під час роботи з фотографіями та в Інтернеті.
- Формат BMP підтримує повну палітру з 16 млн кольорів. Він використовується для зберігання даних без стиснення, тому файли здебільшого мають великий обсяг.
- Формат GIF підтримує палітру всього з 256 кольорів, проте дозволяє зберігати прозорість окремих ділянок зображення та анімацію.

Питання для самоперевірки

1. Що таке комп'ютерна графіка?
2. У чому полягає растрове кодування зображень? Які його переваги та недоліки?
3. До яких зображень застосовують растрове кодування?
4. Для роботи з якими зображеннями призначено графічний редактор GIMP?
5. Опишіть, як користуватися інструментом Пензель.
6. Які особливості роботи з інструментом Заповнення?
7. Як вилучити зайві фрагменти малюнка?
8. Опишіть особливості форматів файлів растрових зображень.

9. Яка особливість вирізняє формат GIF серед розглянутих растрових форматів?
10. Як зберегти зображення у файлі формату JPG?

Вправа 1

▶▶ Створити забраження інструментами малювання GIMP за зразком (рис. 1.8).

1) Запустіть графічний редактор GIMP. Установіть білий колір тла та розпочніть створення малюнка завдовжки 800 і завширшки 600 пікселів.
2) Круглим пензлем із розмитими краями (розмір — близько 450) зобразіть темно-синє небо, а нижню частину замалюйте темно-зеленим кольором.

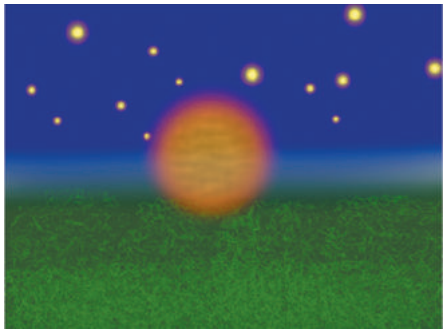


Рис. 1.8

3) Пензлем Sparks додайте зображення зірок.
Якщо відразу це зробити не вдасться, скасуйте останню дію (меню Зміни → Скасувати), змініть параметри Розмір і Проміжок та спробуйте ще раз.
Круглим пензлем великого розміру зобразіть Місяць.
4) Виберіть пензель Grass, доберіть розмір, колір і намалюйте текстуру трави. Збережіть файл з іменем night.xcf.
5) Експортуйте зображення у файл з іменем night.png.
6) Експортуйте зображення у формат BMP. Порівняйте розміри отриманих файлів.

Комп'ютерне тестування

Виконайте тестове завдання 1 із автоматичною перевіркою результату.

§ 2. Багатошарові растрові зображення

Навіть у найпростіших випадках не завжди вдається зразу правильно побудувати малюнок. Якщо ж малюнок складається з багатьох частин, то роботу з ним значно спрощує використання шарів. При цьому зображення будується так, ніби його частини намальовано на окремих прозорих плівках — шарах, які накладаються один на одного. Це дозволяє змінювати в малюнку кожен шар незалежно від інших шарів.

Діалогове вікно Шари

Для виконання різних операцій над шарами призначене меню Шар, а також діалогове вікно Шари (рис. 2.1). Щоб побачити його, слід клацнути вкладку (1).

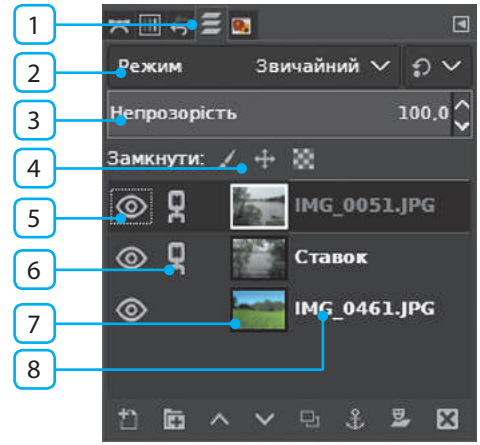


Рис. 2.1

Ми бачимо, що зображення має три шари. Активним є шар IMG_0051.JPG. Він перший у списку, тому затулятиме решту. Щоб **зробити активним інший шар**, достатньо клацнути назву (8) або мініатюру (7) у списку.

Щоб **змінити порядок розташування шарів**, їх треба перетягти у списку вгору або вниз.

Список режимів (2) і регулятор непрозорості (3) діють так само, як і для інструментів малювання. Кнопка (5) перемикає видимість відповідного шару, а позначка (6) означає, що цей шар зв'язаний з іншим шаром, який має таку саму позначку. Для зв'язаних шарів переміщення, змінення розмірів, повороти тощо виконуються одночасно.

Щоб скасувати зв'язок шару з іншими шарами, достатньо клацнути відповідну позначку в діалоговому вікні.

Кнопками (4) можна заблокувати шар від випадкових змін:  — від малювання;  — переміщення;  — стирання.

Унизу вікна містяться кнопки операцій над активним шаром:  — створити новий шар над активним; , — підняти (опустити) на один рівень;  — дублювати, тобто створити ще один такий самий шар;  — вилучити активний шар.

Відкриття зображення з файла

Щоб **відкрити для редагування малюнок з файла**, потрібно вибрати команду меню Файл → Відкрити... . З'явиться стандартне діалогове вікно відкриття файла, у якому слід:

1) відшукати і відкрити папку з файлами;
2) виділити один або кілька файлів;
3) клацнути кнопку Відкрити — над робочим полем з'являться вкладки відкритих зображень, а в робочому полі — одне з них.

Зображення, з яким наразі працює користувач, називають **поточним.**

Відкрити файл також можна перетягуванням значка файла з папки на панель інструментів GIMP. Якщо значок файла перетягти з папки в *робоче поле* з відкритим зображенням, то зображення з файла додасться до поточного як *новий шар*. Після відкриття файла в зображення можна внести зміни та зберегти.

1 На знімку пам'ятки воєнної архітектури XIII–XV ст. Аккерманської (Білгород-Дністровської) фортеці небо вийшло недостатньо виразним. Щоб додати зображення неба з іншого знімка, потрібно:

1) відкрити файл із зображенням неба (рис. 2.2, *а*);
2) перетягти з папки в робоче поле значок файла із зображенням фортеці (рис. 2.2, *б*) — створено новий шар;
3) на шарі із зображенням фортеці, що є активним, стерти зображення неба (рис. 2.2, *в*).

Результат виконаних дій наведено на рис. 2.2, *в*.

а б в

Рис. 2.2

! Якщо ділянки шару під час стирання не стають прозорими, для нього слід увімкнути прозорість (меню Шар → Прозорість → Додати альфа-канал).

Для переміщення шарів один відносно одного призначено інструмент Переміщення  . На панелі параметрів можна вибрати, який шар слід перемістити: активний чи той, до якого належить піксель під вказівником.

Інструменти виділяння

Під час роботи із зображенням у GIMP важливо вміти виділяти потрібну частину. Саме на виділеній ділянці можна малювати, застосовувати заповнення, змінювати форму, колір тощо.

Ви вже виділяли фрагменти малюнка в простому графічному редакторі. Подібні інструменти є також у GIMP, проте вони значно досконаліші. Опис деяких із них подано в таблиці:

Значок	Інструмент	Опис
	Вибір прямокутником	Дозволяють виділяти фрагмент зображення прямокутної або овальної форми. Шляхом перетягування будується приблизний контур виділення (див. далі), який потім уточнюється перетягуванням країв і кутів
	Вибір еліпсом	

Закінчення таблиці

Значок	Інструмент	Опис
	Вільний вибір	Використовується для виділяння фрагментів складної форми. Прямі межі будують послідовним клацанням, а при натиснутій кнопці миші вимальовують криволінійні. Після натискання клавіші **Enter** лінія замикається і з'являється контур виділення
	Вибір пов'язаної ділянки	Після клацання на зображенні виділяється суцільна ділянка, що має колір, схожий на колір початкової точки. Діапазон потрібних кольорів задається регулятором **Поріг** на панелі параметрів
	Вибір за кольором	Працює подібно до попереднього інструмента, але схожі кольори виділяються на всьому зображенні
	Вибір переднього плану	Дозволяє швидко виділити фрагмент складної форми, колір якого помітно відрізняється від тла (наприклад, жовту квітку на тлі зеленого листя)

Результатом використання кожного з інструментів є **контур виділення** — рухлива штрихова лінія, що обмежує виділену частину зображення.

2 Відокремимо зображення квітки (рис. 2.3, *а*) від тла за допомогою інструмента Вибір переднього плану. Для цього нам потрібно:

1) вибрати інструмент  , приблизно обвести за допомогою миші квітку й натиснути клавішу Enter — обведену ділянку буде затінено напівпрозорим заповненням (рис. 2.3, *б*);
2) позначити вказівником миші кольори, що належать квітці (рис. 2.3, *в*). За потреби розмір пензля можна змінити регулятором Ширина обведення на панелі параметрів;
3) клацнути кнопку Вибрати (рис. 2.3, *в*) — напівпрозоре заповнення зникне, а зображення квітки буде виділено.

Виділену ділянку можна скопіювати (меню Зміни → Копіювати), а потім створити нове зображення (меню Файл → Одер-

жати → З буфера обміну) (рис. 2.3, *г*) або вставити на це саме або інше зображення (меню Зміни → Вставити).

а б в г

Рис. 2.3

Вставлений фрагмент розташовується на тимчасовому шарі з назвою Рухомий вибір (рис. 2.4). Після цього необхідно вибрати спосіб його додавання до зображення та клацнути одну з кнопок: — створити з доданого новий шар; — об'єднати додане з поточним шаром.

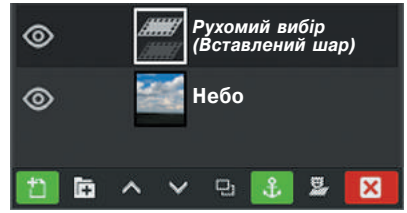


Рис. 2.4

Інструменти заповнення

Для зафарбовування ділянок зображення використовують інструменти Заливання і Градієнт.

Інструмент Заливання дозволяє зафарбувати ділянку зображення кольором переднього плану, кольором тла або текстурою (малюнком).

На панелі параметрів слід вибрати потрібний *тип заповнення* (рис. 2.5). Якщо вибрати варіант Текстурою, то після клацання кнопки зі зразком можна вибрати одну з текстур.

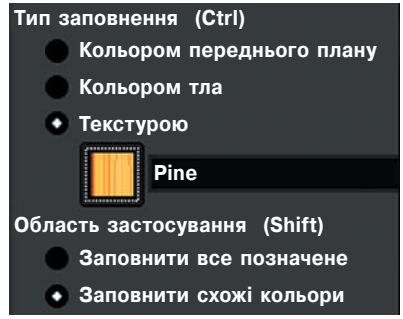


Рис. 2.5

Далі слід вибрати область застосування заповнення:

- Заповнити все позначене — буде зафарбовано позначену ділянку, а якщо не позначено нічого, зафарбується весь малюнок.

- Заповнити схожі кольори — буде зафарбовано лише суцільну ділянку такого кольору, як у початковій точці, або подібного. Ступінь подібності кольорів можна задати регулятором Поріг, який міститься на панелі параметрів.

Інструмент Градієнт  призначено для зафарбовування із плавним переходом від одного кольору до іншого. Набір кольорів і форму градієнта можна вибрати на панелі параметрів.

Інструменти перетворення

До поточного шару або виділеного фрагмента зображення можна застосувати інструменти перетворення — ними змінюють форму і розміри зображення. Щоб **застосувати певний інструмент**, потрібно вибрати шар (фрагмент) і клацнути його, а потім виконати дії, описані в таблиці.

Для керування деякими інструментами відкривається допоміжне вікно, у якому можна ввести додаткові параметри і підтвердити дію.

Значок	Інструмент	Опис
	Обертання	Перетягуванням виконує поворот на потрібний кут. Можна перетягти центр повороту (опорну точку)  в потрібне місце. Дію слід підтвердити кнопкою **Обернути** в допоміжному вікні
	Масштаб	Перетягуванням кутових або бічних маркерів змінює розміри. Позначка  означає, що ширина і висота змінюватимуться разом. Якщо її клацнути, вона набуде вигляду  і можна буде змінювати кожен розмір окремо. Дію слід підтвердити кнопкою **Масштабувати**
	Нахил	Перетягуванням здійснює нахиляння по горизонталі або вертикалі. Дію слід підтвердити кнопкою **Нахилити**
	Дзеркало	Віддзеркалює зображення. Потрібно вибрати напрямок (горизонтально чи вертикально) на панелі параметрів інструмента і клацнути зображення

Питання для самоперевірки

1. Поясніть, як утворюється багатошарове зображення.
2. Опишіть призначення елементів діалогового вікна Шари.
3. Як додати до зображення порожній шар?
4. Опишіть особливості інструментів виділяння фрагмента.
5. Як використовують інструмент Заповнення?
6. Що таке градієнт?
7. Як відкрити малюнок із файла для редагування в GIMP?
8. Як додати до зображення шар із малюнком із файла?

Вправа 2

▶▶ Побудувати колаж з елементів, які розташовуються на кількох шарах.

1) Запустіть графічний редактор GIMP. Відкрийте зображення з файла з іменем leaf.png (рис. 2.6, *а*).
2) Виділіть зображення листка і скопіюйте в буфер обміну.
3) Створіть нове зображення розміром 1000 × 1000 пікселів.
4) Вставте зображення листка з буфера обміну на новий шар (кнопка ). Продублюйте цей шар 5 разів.
5) Інструментами перетворення побудуйте колаж (рис. 2.6, *б*).
6) Додайте на окремому шарі тло з градієнтним заповненням. Збережіть результат у файл з іменем leaves.xcf.

а

б

Рис. 2.6

Комп'ютерне тестування

Виконайте тестове завдання 2 із автоматичною перевіркою результату.

§ 3. Векторна графіка

У багатьох випадках під час створення малюнка зручніше зазначати не колір його пікселів, а властивості та розміщення елементів (відрізків, многокутників, криволінійних фігур). Саме таким чином кодуються зображення у векторній графіці.

Векторні зображення

Розглянемо зображення іграшкового автомобіля (рис. 3.1). Як бачимо, воно складається з трьох кіл, двох прямокутників, чотирикутника і п'ятикутника.

Опишемо властивості одного з об'єктів: прямокутник завдовжки 34 мм і завширшки 20 мм, колір заповнення — 1 (див. таблицю на с. 10), колір штриха — 4, товщина штриха — 1 мм, координати лівого верхнього кута (50; 120). Якщо таким чином описати всі фігури, то отримаємо код векторного малюнка.

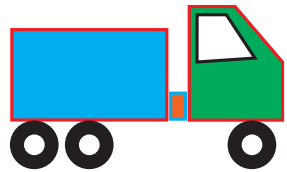


Рис. 3.1

Векторна графіка має як переваги, так і недоліки. *Основною перевагою* є те, що в разі змінення розмірів і обертання якість малюнка зберігається. Серед *недоліків* — складність у побудові реалістичного зображення, адже для цього доводиться використовувати дуже багато об'єктів.

Для роботи з векторними зображеннями є багато програм. Найчастіше застосовують програми Inkscape, CorelDraw, Adobe Illustrator, LibreOffice Draw (рис. 3.2) та ін.

Inkscape

CorelDraw

Adobe Illustrator

LibreOffice Draw

Рис. 3.2

Властивості векторних зображень

Розмір файла векторного зображення залежить від кількості окремих об'єктів, з яких це зображення утворено.

Під час формування векторного зображення можна використовувати не суцільні кольори, а градієнти — плавні переходи від одного кольору до іншого. Розмивання окремих елементів, імітація тіні та інші засоби дають змогу отримати майже реалістичні малюнки (рис. 3.3).

Рис. 3.3

Формати файлів векторних зображень

Для запису векторних зображень у файл існує багато форматів. Розглянемо деякі з них.

Формат SVG (англ. *Scalable Vector Graphics* — масштабована векторна графіка) підтримується багатьма програмами, зокрема графічним редактором Inkscape.

Нерідко популярні векторні графічні редактори мають власні формати файлів: наприклад, CorelDRAW — CDR, а Adobe Illustrator — AI та ін.

Формати CGM, WMF, EPS підтримують як векторну, так і растрову графіку.

Векторний графічний редактор Inkscape

Векторні зображення можна створювати в різних програмах, зокрема й у текстовому процесорі Word. Проте найбільше можливостей надають векторні графічні редактори, наприклад Inkscape.

Графічний редактор Inkscape вільно розповсюджується та має багато засобів для опрацювання векторних зображень. Програма є багатоплатформною, тобто одночасно виходять версії для різних операційних систем (Windows, Linux тощо).

Відразу після запуску графічного редактора відкривається його головне вікно. Розглянемо його детальніше.

Крім звичних вам рядків заголовка та меню, ми побачимо робоче поле з чистим «аркушем» та інші засоби керування (рис. 3.4).

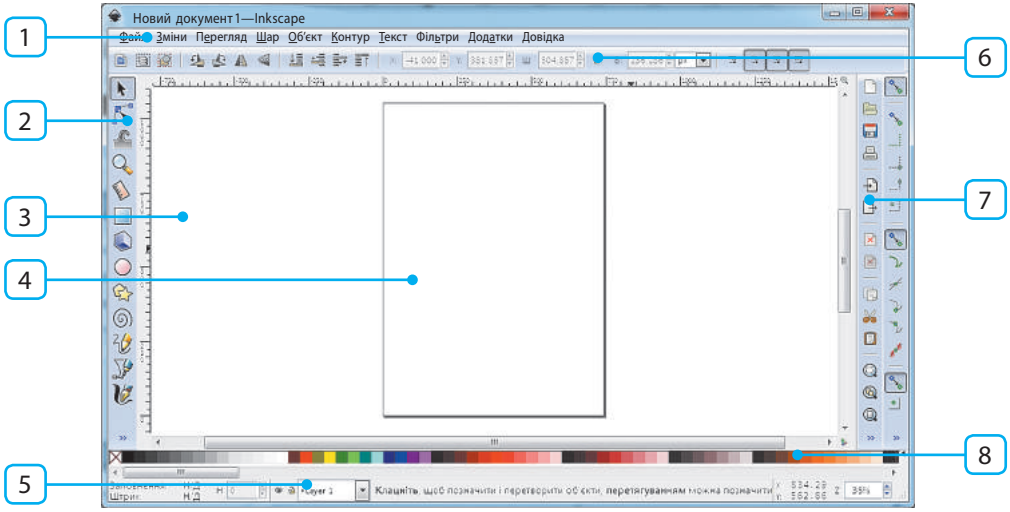


Рис. 3.4

1 — рядок меню
2 — панель інструментів
3 — робоче поле
4 — аркуш
5 — рядок стану
6 — панель параметрів інструментів
7 — панель команд
8 — палітра

Побудова фігур

Об'єкти, з яких будується векторне зображення, поділяються на **фігури**, які мають правильну форму, і **криві**, форму яких можна змінювати довільно.

Більшість кнопок, розміщених на панелі інструментів, призначено для побудови графічних об'єктів. Якщо клацнути кнопку певного інструмента, то на панелі параметрів інструментів з'являться значення його параметрів. У разі подвійного клацання відкриється вікно для додаткових налаштувань.

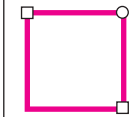
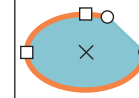
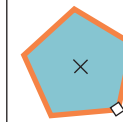
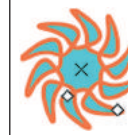
Розглянемо інструменти, призначені для побудови фігур: Прямокутник, Еліпс, Многокутник і Спіраль.

Щоб **додати фігуру** за допомогою одного з цих інструментів, потрібно виконати такі дії:

1) клацнути кнопку потрібного інструмента на панелі;
2) перемістити вказівник на робоче поле;
3) натиснути ліву кнопку миші і, не відпускаючи, пересунути вказівник в інше місце та відпустити кнопку миші.

Форму створеної фігури можна змінити перетягуванням маркерів — білих квадратиків або кружечків.

Розглянемо дію маркерів фігур різних типів:

Значок	Інструмент	Дія маркерів	Приклад
	Прямокутник	Якщо перетягувати круглі маркери (спочатку вони можуть бути суміщені), то отримаємо округлені кути, якщо квадратні — змінимо розміри фігури	
	Еліпс	Якщо круглий маркер перетягнути від центра, то «виріжемо» сектор, до центра — отримаємо сегмент. Квадратні маркери — для зміни розмірів	
	Многокутник	Завдяки додатковим параметрам можна будувати різні многокутники: правильні й неправильні, опуклі й зірчасті, з округленими кутами та ін. Опуклий многокутник має один маркер (ним можна змінювати розмір і обертати), зірчастий — два маркери (один для зміни внутрішнього радіуса і форми, інший — для зміни зовнішнього радіуса й обертання)	 
	Спіраль	Побудована спіраль має на кінцях два маркери, обертаючи які за допомогою миші можна додавати або вилучати витки ззовні та всередині	

Утримування натиснутими клавіш Ctrl, Shift або Alt дає додаткові можливості при змінюванні форми об'єктів. Їх дію пояснюють підказки, що з'являються внизу екрана.

Операції над об'єктами

Перш ніж виконувати певну операцію над об'єктами, їх потрібно виділити, скориставшись інструментом Стрілка.

Щоб **виділити окремий об'єкт**, на ньому слід клацнути. Навколо об'єкта з'явиться пунктирна прямокутна рамка, а біля неї — стрілки для змінення розмірів (рис. 3.5).

Щоб **виділити декілька окремих об'єктів**, їх слід по черзі клацнути, утримуючи натиснутою клавішу Shift.

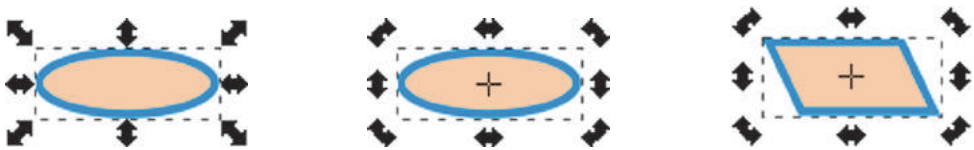
Над виділеними об'єктами можна виконувати певні операції.

Щоб **перемістити об'єкт**, його слід перетягнути мишею.

Для **масштабування (змінення розмірів) об'єкта** треба перетягнути відповідні стрілки (рис. 3.5, *а*): *кутовими* стрілками можна змінити одночасно довжину і ширину, а *бічними* — лише один із розмірів.

Об'єкти можна *обертати* та *нахиляти*. Якщо клацнути на виділеному об'єкті, то стрілки змінять вигляд і з'явиться хрестик, що позначає центр обертання (рис. 3.5, *б*). Під час перетягування кутових стрілок об'єкт обертатиметься. Центр обертання також можна перетягнути в інше місце.

Аналогічно, за допомогою стрілок біля сторін прямокутника, виконується операція нахиляння об'єкта (рис. 3.5, *в*). Під час обертання та нахиляння об'єкта центр залишається нерухомим.



а б в

Рис. 3.5

Кнопки на панелі параметрів інструмента Стрілка дають змогу швидко змінити положення об'єкта:  ,  — повернути об'єкт на 90° за ходом годинникової стрілки або проти;  ,  — відобразити об'єкт відносно горизонтальної або вертикальної осі, що проходить через центр обертання.

Штрих і заповнення об'єктів

Під час малювання можна змінити кольори штриха і заповнення об'єктів. Для цього потрібно вибрати команду меню Об'єкт → Заповнення та штрих або на панелі команд натиснути кнопку  . Відкриється вікно Заповнення та штрих (рис. 3.6).

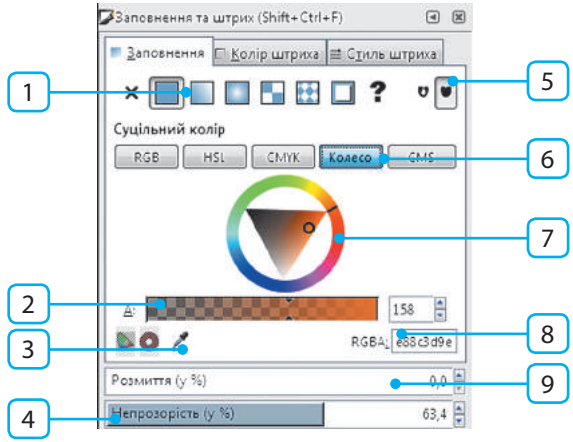


Рис. 3.6

1 — кнопки вибору типу заповнення
2 — регулятор прозорості кольору
3 — піпетка для вибору і копіювання певного кольору
4 — регулятор прозорості об'єкта
5 — кнопки вибору способу заповнення складних об'єктів
6 — перемикання режиму вибору кольору
7 — колесо кольорів
8 — числовий код кольору (у шістнадцятковій системі)
9 — регулятор розмиття об'єкта

Повернемося до рис. 3.6. У вікні Заповнення та штрих подано вкладку Заповнення під час вибору **кольору для суцільного заповнення** з використанням колеса кольорів (7).

Як і в растровому редакторі GIMP, щоб вибрати колір, потрібно, клацнувши кільце, вибрати тон, а потім, клацнувши трикутник, вибрати відтінок кольору.

Щоб заповнення фігури було напівпрозорим, слід клацнути відповідне місце регулятора під кільцем. Якщо вікно не закриває малюнок, то всі зміни кольору буде видно на екрані.

Розглянемо основні елементи керування:

Кнопка	Тип заповнення	Приклад
×	Заповнення відсутнє, видно лише штрих	
	Суцільне заповнення вибраним кольором з урахуванням прозорості	
	Лінійний градієнт — плавний перехід від одного кольору до іншого або від зафарбованої ділянки до прозорої	
	Радіальний градієнт — плавний перехід кольорів від центра до країв фігури	
	Візерунок	

На вкладці Колір штриха таким самим чином можна вибирати колір штриха об'єкта.

На вкладці Стиль штриха можна налаштувати товщину штриха у вибраних одиницях (наприклад, у міліметрах) та інші параметри (рис. 3.7).

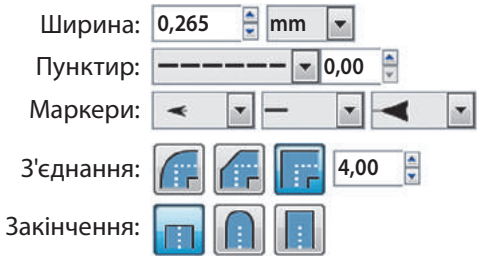


Рис. 3.7

Особливості побудови й опрацювання векторних зображень

Створення графічних зображень, як і звичайне малювання, — процес творчий. Починати треба з простих зображень, які містять небагато окремих елементів.

Щоб **вилучити фрагмент малюнка**, його треба вибрати інструментом Стрілка і натиснути клавішу Delete. Це не обов'язково робити зразу: усі частини векторного зображення є окремими об'єктами, їх можна змінити або вилучити в будь-який момент.

Краще роздивитися дрібні деталі допоможе керування масштабом за допомогою кнопок, що на панелі команд: — показати вибрані об'єкти, — показати весь малюнок, — показати весь аркуш.

> **!** Зміна масштабу жодним чином не впливає на малюнок, а змінюється лише його розмір на екрані.

Робота з файлами в Inkscape

Як і багато інших графічних редакторів, Inkscape дозволяє зберігати й відкривати для опрацювання файли різних форматів. Проте якщо роботу з малюнком не закінчено, то під час збереження слід вибрати тип файла Файл Inkscape SVG.

Для роботи з документами на панелі команд зібрано кнопки: — створити новий документ; — надрукувати документ; — відкрити наявний документ; — зберегти документ.

Щоб **записати зображення у файл у растровому форматі**, потрібно:

1) вибрати команду меню Файл → Експортувати як зображення PNG;
2) у діалоговому вікні задати параметри експорту й ім'я файла;
3) натиснути кнопку Експортувати.

Питання для самоперевірки

1. Поясніть суть векторного кодування зображень.
2. Опишіть порядок побудови прямокутника.
3. Як змінити розмір побудованої фігури?
4. Як повернути об'єкт на певний кут?
5. Поясніть, як користуватися колесом кольорів, що міститься у вікні Заповнення та штрих.

6. Як зробити заповнення фігури напівпрозорим?
7. Як змінити товщину штриха фігури?
8. Як зберегти малюнок у файлі?
9. Які кнопки керування масштабом ви знаєте?
10. Який формат файла потрібно вибрати, щоб зберегти зображення, над яким не завершено роботу?

Вправа 3

Побудувати зображення, що складається з фігур.

1) Запустіть графічний редактор Inkscape. Розгорніть вікно на весь екран.
2) Знайдіть панелі інструментів, команд, параметрів інструментів, палітру.
3) Установіть такий масштаб, щоб було видно цілу сторінку. Скористайтесь кнопкою .
4) Побудуйте позначені елементи зображення в порядку, наведеному на рис. 3.8, за описом:

Рис. 3.8

1 (*поверхня землі*) — прямокутник сірого кольору. Клацніть його і нахиліть, перетягнувши стрілку біля сторони (с. 27);
2 (*стіна будинку*) — прямокутник;
3 (*стовбур дерева*) — два трикутники. Скористайтесь інструментом Многокутник;
4 (*крона дерева*) — кілька кругів із зеленим заповненням і білим кольором штриха;
5 (*сонечко*) — многокутник. Зірчасту форму увімкніть кнопкою на панелі параметрів інструмента;
6 (*хмарка*) — кілька напівпрозорих еліпсів.

5) Застосуйте розмиття до еліпсів (хмарки) і трохи змініть кольори деяких із них.
6) Додайте ще кілька елементів малюнка на свій розсуд. Збережіть зображення у файлі з іменем picture.svg.

Комп'ютерне тестування

Виконайте тестове завдання 3 із автоматичною перевіркою результату.

§ 4. Криві. Робота з контурами

Більшість векторних малюнків, що ви бачите у книжках і журналах, складаються не лише з простих геометричних фігур, про які вже йшлося, а й зі складних за формою елементів (рис. 4.1).

Рис. 4.1

Криві Безьє

Найменшим елементом векторного зображення є лінія особливого виду — **крива Безьє**. Математична формула кривої Безьє доволі складна, а от її форма залежить від розташування всього чотирьох точок: початкової, кінцевої і двох керувальних (рис. 4.2).

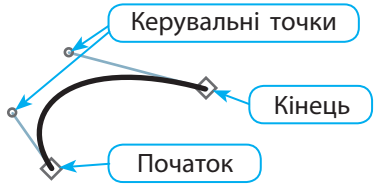


Рис. 4.2

Якщо одну керувальну точку сумістити з початковою, а іншу — з кінцевою, то крива Безьє набуде вигляду відрізка прямої.

Французький учений, математик та інженер П'єр Безьє розробив ці криві у 1968 році, проектуючи кузов автомобіля для компанії «Рено» (Франція).

В основі будь-якого об'єкта векторного зображення лежить **контур**, що складається з однієї або більше кривих Безьє (**сегментів**), сполучених у точках, які називають **вузлами**.

1 Еліпс з вирізаним сектором, який побудовано за допомогою інструмента Еліпс (рис. 4.3), складається з шести сегментів, два з яких є відрізками прямих. Під час побудови і подальших операцій цього не видно, тому форму фігури не можна змінювати довільно.

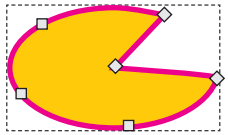


Рис. 4.3

Кривими називають об'єкти, в яких, на відміну від фігур, є можливість вільно змінювати форму, пересуваючи вузли та керувальні точки.

Інструменти для побудови кривих

Розглянемо інструменти Олівець і Перо, якими малюють криві.

Інструмент Олівець призначений для малювання ліній довільної форми і відрізків.

Щоб **намалювати криву**, потрібно:

1) вибрати інструмент Олівець, натиснути ліву кнопку миші та, не відпускаючи її, рухом миші малювати лінію;
2) відпустити кнопку миші, щоб закінчити малювання кривої. Якщо початок і кінець кривої збігаються, вона замкнеться.

Щоб **намалювати відрізок**, потрібно вибрати інструмент і послідовно клацнути мишею в початковій та кінцевій точках.

Інструмент Перо призначений для малювання кривих із прямих і криволінійних сегментів.

2 Намалюємо инструментом Перо замкнену криву у формі підкови (рис. 4.4). Для цього потрібно клацнути точки 1 і 2, перетягнути вправо точку 3, клацнути точки 4 і 5, перетягнути вліво точку 6 і знову клацнути точку 1.

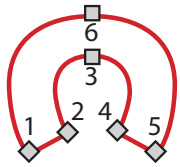


Рис. 4.4

Щоб отримати незамкнений контур, у його останній точці слід клацнути двічі.

Робота з контурами

Інструмент Вузол  призначений для змінювання контурів об'єктів. Вибравши і клацнувши одну з кривих, ми побачимо на ній або поряд із нею квадратики, що позначають вузли.

Для виконання більшості операцій вузли і сегменти слід виділяти. Щоб виділити *вузол*, на ньому потрібно клацнути, щоб виділити *кілька вузлів* — клацнути, утримуючи клавішу Shift. Щоб виділити *сегмент*, потрібно виділити вузли на його кінцях.

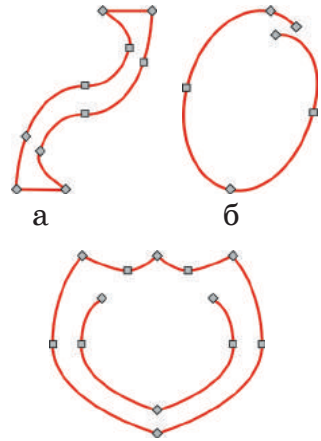


Рис. 4.5

Розрізняють прості і складені контури. **Простий контур** може бути *замкненим* (рис. 4.5, *а*) або *незамкненим* (рис. 4.5, *б*). У незамкненому контурі два вузли розташовуються на його кінцях, а в кожному з решти вузлів сполучаються по два сегменти.

Складений контур має дві або більше частин, не сполучених між собою (рис. 4.5, *в*).

Змінення форми контура

Існує кілька способів змінювати форму контура:

- якщо перетягнути один або декілька вузлів у інше місце, то зміниться форма прилеглих до них сегментів;
- якщо перетягнути будь-яку точку контура, то зміниться форма сегмента, якому належить ця точка.

Біля виділеного вузла з'являться один або два відрізки, дотичні до кривої, кожен із керувальною точкою (круглим *маркером*) на кінці. Маркери також з'являться у двох сусідніх вузлів (рис. 4.6). Перетягуючи маркери, можна змінювати форму сегментів. Якщо у виділеного вузла не з'явилися маркери, це означає, що вони з ним суміщені. Їх можна «витягнути» мишею, утримуючи клавішу Shift.

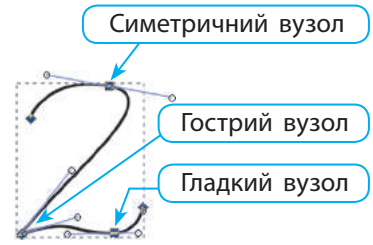


Рис. 4.6

Щоб довільно змінювати контури фігур (прямокутників, кругів, многокутників тощо), їх слід виділити й вибрати команду меню Контур → Об'єкт у контур.

Робота з вузлами

На панелі параметрів інструмента Вузол містяться кнопки додаткових операцій із вузлами: — додати вузли посередині виділених сегментів; (або клавіша Delete) — вилучити виділені вузли (контур залишиться нерозривним, якщо він був таким до цього); — з'єднати два кінцеві вузли, замінивши їх одним вузлом; — з'єднати два кінцеві вузли новим сегментом.

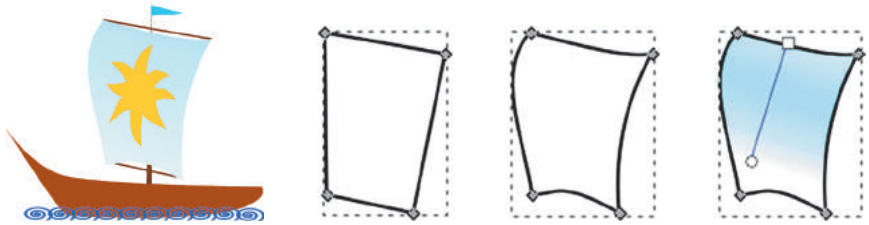
Якщо треба **з'єднати кінці двох або більше контурів**, ці контури спочатку слід об'єднати в один об'єкт — виділити і вибрати команду меню Контур → Об'єднати.

Щоб **змінити тип вузла**, його треба вибрати і клацнути одну з кнопок: — зробити вузол гострим; — зробити вузол гладким; — зробити вузол симетричним.

3 Намалюємо вітрила для кораблика (рис. 4.7, *а*).

Для цього нам потрібно виконати такі дії.

1) Інструментом Перо клацнути по черзі у вершинах чотирикутника (рис. 4.7, *б*). Щоб контур став замкненим, першу вершину в кінці побудови треба клацнути ще раз.
2) Інструментом Вузол вибрати чотирикутник, а потім перетягуванням точок сегментів надати контуру бажаної форми (рис. 4.7, *в*).



а б в г

Рис. 4.7

3) У вікні Заповнення та штрих вибрати блакитний колір, потім задати вид заповнення лінійний градієнт і налаштувати його, перетягнувши керувальні точки та дібравши кольори (рис. 4.7, *г*).

Питання для самоперевірки

1. Опишіть особливості кривої Безьє.
2. Якими інструментами будують криві?
3. Які операції виконують інструментом Вузол?
4. Як додати на контурі новий вузол; вилучити зайвий вузол?
5. Які є типи вузлів?
6. Який контур називають складеним?
7. Які є типи сегментів?

Вправа 4

Побудувати зображення зі складеним контуром.

1) Запустіть графічний редактор Inkscape. Користуючись інструментом Перо, побудуйте замкнену ламану (рис. 4.8, *а*), послідовно клацаючи її вершини.
2) Виберіть інструмент Вузол і клацніть ламану. Зробіть частину вузлів гладкими: виділіть їх і, користуючись кнопкою , отримайте потрібну форму голови (рис. 4.8, *б*).
   За потреби перетягніть маркери вузлів.
3) Доберіть кольори штриха й заповнення, приблизно як на рис. 4.8, *б*.
4) Домалюйте інструментом Перо рот і носик котика (рис. 4.8, *в*).
5) Домалюйте інструментом Еліпс очі котика.
6) Збережіть зображення у файлі з іменем cat.svg.

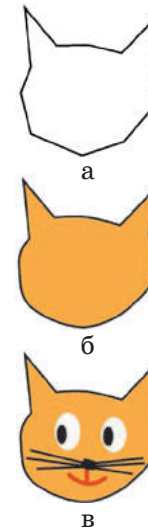


Рис. 4.8

Комп'ютерне тестування

Виконайте тестове завдання 4 із автоматичною перевіркою результату.

§ 5. Копіювання об'єктів

Намалювати багато однакових предметів — непроста справа, бо досягти високої схожості зазвичай складно. Завдяки можливостям графічного редактора створювати однакові об'єкти легко.

Копіювання та клонування об'єктів

Розглянемо рослинний орнамент, який наведено на рис. 5.1. Він складається з повторюваних зображень квітки та листка, тому його зручно створювати шляхом копіювання.

Рис. 5.1

Для виконання операцій з копіями об'єкта існують команди меню Зміни і група кнопок  ,  ,  на панелі команд.

Отримати копію дуже просто: об'єкт треба виділити та вибрати команду меню Зміни → Дублювати або клацнути кнопку Дублювання  на панелі команд. Після цього копію можна перетягнути в потрібне місце.

> ! Відразу після створення копія стає самостійним об'єктом, жодним чином не пов'язаним з оригіналом.

Іноді трапляються випадки, коли потрібно швидко змінити властивості низки об'єктів. Наприклад, змоделювавши візерунок для вишивання хрестиком червоними, зеленими і жовтими нитками, ви вирішили замінити зелений колір на синій. У такому випадку доцільно створити по одному зображенню різнокольорових хрестиків, а далі робити не копії, а клони.

Клон — це копія, яка зберігає зв'язок з оригінальним об'єктом. Його колір, розмір, положення, товщина штрихів, інші властивості змінюються разом з оригіналом.

Для **клонування об'єкта** слід вибрати об'єкт і викликати команду меню Зміни → Клонувати → Створити клон або клацнути кнопку на панелі команд.

Щоб **розірвати зв'язок між клоном і оригіналом**, клон потрібно виділити й клацнути кнопку на панелі команд. Після цього клон стає звичайним об'єктом.

Вирівнювання об'єктів

Об'єкти, з яких будується векторне зображення, зазвичай потребують вирівнювання один відносно одного.

Якщо скопійовані об'єкти відразу не вдається акуратно розмістити на зображенні (рис. 5.2, *а*), то варто скористатися тими можливостями Inkscape, які дозволять їх швидко вирівняти й рівномірно розподілити (рис. 5.2, *б*).

а б

Рис. 5.2

Перед тим як **об'єкти вирівняти**, їх спочатку потрібно виділити, а потім вибрати команду меню Об'єкт → Вирівняти... або скористатися кнопкою на панелі команд.

З'явиться вікно Вирівняти та розподілити... (рис. 5.3), у якому команди проілюстровано відповідними піктограмами. Детальнішу інформацію можна отримати у спливних підказках.

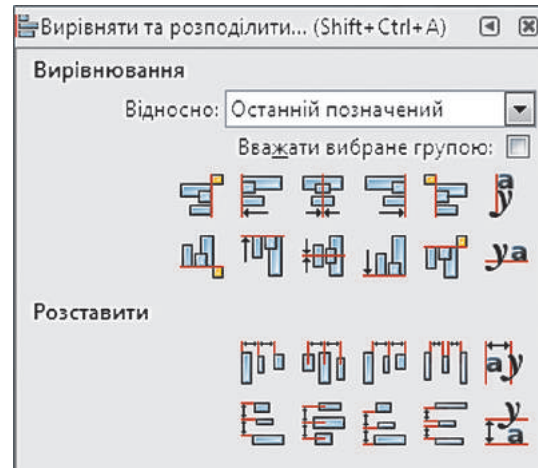


Рис. 5.3

У групі Вирівнювання в полі Відносно потрібно вибрати *об'єкт-якір*, відносно якого в окремих випадках здійснюватиметься вирівнювання. Це може бути як сторінка, так і найбільший або найменший із виділених об'єктів тощо.

Під час виконання команд вирівнювання об'єкт-якір залишається нерухомим, а решта об'єктів відповідним чином зміщується.

У групі Розставити містяться команди для рівномірного розподілення об'єктів у різний спосіб.

Точно розмістити об'єкти один відносно одного допоможе *сітка*, яку вмикають командою меню Перегляд → Сітка сторінки. Тепер у ході перетягування мишею вказівник прилипатиме до найближчого перетину ліній (вузла) сітки.

Параметри сітки (відстань між лініями, колір тощо) можна налаштувати у вікні, що відкривається, командою Зміни → Налаштування на вкладці Інтерфейс → Сітки.

Для **керування прилипанням** до інших елементів (контурів, рамок об'єктів тощо), зокрема для тимчасового вимкнення цього режиму, слід скористатися кнопками панелі керування прилипанням. Якщо на екрані ця панель відсутня, треба вибрати команду меню Перегляд → Показати/Сховати і клацнути назву панелі.

Питання для самоперевірки

1. У яких випадках використовують копіювання об'єктів?
2. Що таке клон? Чим він відрізняється від інших об'єктів?
3. Яке призначення кнопки  на панелі команд?
4. Як вирівняти вибрані об'єкти за верхнім краєм?
5. Як працюють команди вирівнювання за об'єктом-якорем?
6. Як встановити між вибраними об'єктами однакові проміжки по горизонталі?
7. Як розірвати зв'язок між клоном і оригіналом?
8. Яке призначення сітки?
9. Який порядок дій для побудови орнаменту (рис. 5.4)?

Рис. 5.4

Вправа 5

1. Побудувати візерунок для вишивання хрестиком, забезпечити можливість добирати кольори і форму його частин.

1) Запустіть графічний редактор Inkscape. Увімкніть сітку, прилипання вказівника до сітки та вузлів-вершин.
2) Побудуйте три квадрати, розмір яких дорівнює одній клітинці сітки. Задайте їм різні кольори заповнення, вимкніть штрих.
3) Створіть клони квадратів і побудуйте з них візерунок. Користуючись прилипанням, клони вишикуйте рядками. Якщо випадково замість клона у візерунок буде вставлено початковий квадрат, то щоб його відшукати, виділіть один із клонів і натисніть сполучення клавіш Shift + D.
4) Змінюйте колір початкових квадратів у візерунку та слідкуйте за клонами: вони теж мають змінювати колір.
5) Змініть форму одного з квадратів: його клони теж мають змінити форму (рис. 5.5).

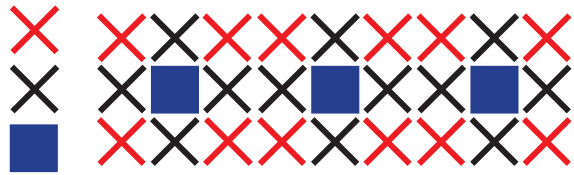


Рис. 5.5

6) Змініть форму інших квадратів, щоб отримати візерунок для вишивання. Збережіть зображення у файлі з іменем ornament.svg. Завершіть роботу за комп'ютером.

2. Знайдіть в Інтернеті урок з покрокового створення мозаїки з клонів об'єкта в графічному редакторі Inkscape та перегляньте його.

Комп'ютерне тестування

Виконайте тестове завдання 5 із автоматичною перевіркою результату.

Практична робота 1
Створення простих векторних зображень

Завдання: побудувати графічне зображення з об'єктів різних типів за зразком, використовуючи операції копіювання і групування об'єктів.

Обладнання: комп'ютер зі встановленим графічним редактором Inkscape.

Хід роботи

Під час роботи з комп'ютером дотримуйтеся правил безпеки.

1. Запустіть графічний редактор Inkscape. Побудуйте зображення равлика за зразком (рис. 1).

Рис. 1

1) Побудуйте спіраль, яка зображує черепашку равлика.
2) Додайте ще одну спіраль (білого кольору, розмиту і без заповнення) для імітування об'ємності черепашки.
3) Побудуйте криву у формі, що нагадує тіло равлика. Якщо прилипання до сітки заважає — вимкніть його. Виправте форму, скориставшись інструментом Вузол.
4) Додайте до зображення інші елементи. Збережіть зображення у файлі з іменем snail.svg.

2. Побудуйте зображення рушника із простим геометричним орнаментом за зразком (рис. 2).

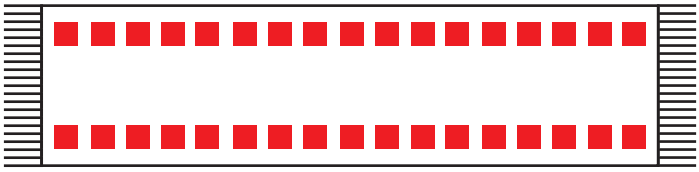


Рис. 2

1) Побудуйте прямокутник — основу для полотна.
2) Побудуйте в межах прямокутника квадратик червоного кольору для створення орнаменту.

3) Зробіть копії квадратика — стільки, скільки потрібно для одного рядка.
4) Розмістіть два квадратики в кутках (приблизно) полотна.
5) Виділіть усі квадратики і вирівняйте їх уздовж горизонтальної прямої на однаковій відстані один від одного.
6) Зробіть копію рядка квадратів і розмістіть біля протилежної сторони прямокутного полотна.
7) Так само побудуйте торочку на кінцях рушника.
8) Збережіть зображення у файлі з іменем rushnik.svg.

Зробіть висновок про результати роботи.

§ 6. Текстові об'єкти

Малюнки часто супроводжуються текстом, бо це додає їм інформативності. Програма Inkscape має широкі можливості для створення написів і оформлення ілюстрованих публікацій.

Додавання тексту до зображення

Написи до малюнка зручно додавати інструментом Текст A. Вибравши інструмент, треба клацнути місце, де має починатися напис (там з'явиться текстовий курсор), та ввести напис за допомогою клавіатури.

При цьому не варто зважати на розмір символів, бо розмір готового напису можна змінювати як завгодно, користуючись інструментом Стрілка. Колір штриха та заповнення літер змінюється так само, як і для інших об'єктів. Також на панелі параметрів інструмента для напису можна змінити гарнітуру, накреслення, вирівняти абзаци тощо.

Виділивши окремий символ, його можна змістити вгору чи вниз або повернути на певний кут відносно інших символів. Для кожного символа окремо можна задати стиль штриха та заповнення (рис. 6.1).

Рис. 6.1

Додаткові можливості для опрацювання напису надає меню Текст. Команда меню Текст → Текст і шрифт відкриває діалогове вікно для більш детальної роботи з написом.

Робота з текстовими об'єктами

Уведений текст можна розмістити вздовж кривої довільної форми (рис. 6.2). Для цього потрібно додати до малюнка текст і криву й виділити їх разом, а потім вибрати команду меню Текст → Розмістити по контуру.

Рис. 6.2

У разі змінення форми кривої буде відповідно змінюватися і форма тексту.

Текстовий об'єкт можна прикрасити. Для цього його потрібно виділити й скористатися командою меню Контур → Зв'язане втягування. За текстовим об'єктом з'явиться ще один об'єкт (рис. 6.3), що слугує тлом. Його форму можна змінювати шляхом перетягування маркера, можна задати кольори штриха та заповнення, відмінні від початкового об'єкта.

Рис. 6.3

У разі змінення тексту буде відповідно змінюватися й об'єкт втягування.

Текстові блоки

Якщо, вибравши інструмент Текст, далі виконати перетягування мишею, то буде створено прямокутний текстовий блок (рис. 6.4). Текст, який буде набрано, розміститься всередині цього блока.

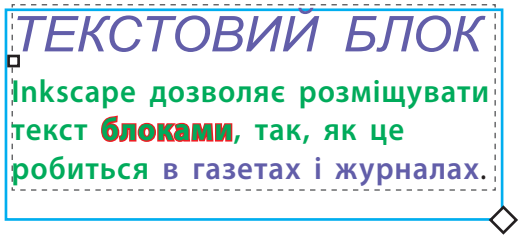


Рис. 6.4

Якщо текст не поміщається в блоці, межа блока набуває червоного кольору. У такому випадку розміри блока можна змінити маркером, який міститься в правому нижньому куті.

Питання для самоперевірки

1. Для чого призначений інструмент Текст?
2. Як змінити колір штриха, яким обведено літери в тексті?
3. Як створити текстовий об'єкт, який наведено на рис. 6.5?
4. Що означає команда Зв'язане втягування?
5. Як створити текстовий блок?
6. У якому випадку межа текстового блока має червоний колір?

Рис. 6.5

Вправа 6

Створити текстовий об'єкт, додати напис і застосувати зв'язане втягування.

1) Запустіть графічний редактор Inkscape. Виберіть інструмент Текст A, клацніть потрібне місце та, після появи курсора, введіть назву свого навчального закладу.
2) Побудуйте текстовий об'єкт: уведіть власні прізвище та ім'я.
3) Виберіть інструмент Перо, побудуйте криву з 4 вузлів.
4) Виділіть криву разом із текстовим об'єктом і виберіть команду меню Текст → Розмістити по контуру. Виберіть інструмент Вузол і змініть форму кривої на свій розсуд.
5) Створіть напис за зразком (рис. 6.6). Додайте до напису зв'язане втягування (рис. 6.7).

Рис. 6.6

Рис. 6.7

6) Доберіть для об'єкта кольори штриха та заповнення. Збережіть результат роботи у файлі з іменем text.svg.

Комп'ютерне тестування

Виконайте тестове завдання 6 із автоматичною перевіркою результату.

§ 7. Складені векторні зображення

Під час створення малюнка, що складається з різних об'єктів, часто виникає потреба в опрацюванні кількох об'єктів як одного цілого. У редакторі Inkscape є засоби, які дозволяють це зробити.

Об'єднання контурів

На рис. 7.1 зображено чотири прямокутники, що мають спільне градієнтне заповнення. Такого ефекту можна досягти, просто вибравши всі прямокутники і скориставшись інструментом Градієнт. Але для того щоб потім його можна було регулювати, спочатку потрібно **об'єднати контури** відповідною командою меню Контур ( ).

Рис. 7.1

Об'єднані контури далі опрацьовують як єдиний складений контур, тобто вони вже мають спільні колір штриха і заповнення.

Щоб **розділити складений контур на окремі частини**, потрібно виділити об'єкт, якому він належить, і вибрати команду меню Контур → Розділити ( ).

Групування об'єктів

Для **групування об'єктів** треба вибрати всі об'єкти, що ввійдуть у групу, і вибрати команду меню Об'єкт → Згрупувати або скористатися кнопкою панелі команд.

Згруповані об'єкти можна пересувати й масштабувати як єдине ціле. При цьому кожен з них має власні кольори та стилі штриха і заповнення. Під час групування об'єктів єдиний контур не утворюється.

Щоб **змінити контур, штрих, заповнення** одного з об'єктів, слід увійти в групу, двічі її клацнувши, внести зміни та вийти з групи, двічі клацнувши поза її межами.

У разі потреби об'єкти можна **розгрупувати** відповідною командою меню Об'єкт або кнопкою .

Упорядкування перекриття об'єктів

Об'єкти, з яких складається зображення, іноді можуть перекривати один одного. Об'єкт, який створено останнім, розміщується найвище і закриває всі попередні. Працюючи над складним малюнком, не завжди вдається передбачити послідовність появи об'єктів. Унаслідок цього частини об'єкта, які повинні бути повністю видимими, закриваються.

Щоб **змінити порядок розташування об'єктів**, потрібно інструментом Стрілка вибрати об'єкти, які слід перемістити. Далі на панелі параметрів інструмента потрібно клацнути потрібну кнопку: — підняти вибране на один рівень; — опустити вибране на один рівень; — опустити вибране на задній план; — підняти вибране на передній план.

1 Чи може місяць затуляти хмари (рис. 7.2, а)? Щоб виправити перекриття зображених об'єктів, виберемо об'єкт, що зображує місяць, і клацнемо кнопку (опустити на задній план) (рис. 7.2, *б*).

а б

Рис. 7.2

Багатошарові векторні зображення

Під час роботи зі складними зображеннями споріднені об'єкти зручно розміщувати в окремих **шарах** — їх можна уявити як прозорі аркуші, накладені один на одного.

На кожному шарі розташовуються певні об'єкти (рис. 7.3).

Рис. 7.3

Застосування шарів дозволяє вимикати видимість частини зображення, блокувати об'єкти шару від випадкових змін тощо.

Суміщенням таких шарів отримується цілісне зображення, що називається **багатошаровим.**

У меню Шар зібрано команди для додавання і перейменування, показу й приховання, блокування і розблокування, зміни взаємного розміщення шарів, а також переміщення виділених об'єктів з одного шару в інший.

У рядку стану міститься список шарів, а також кнопки керування видимістю й блокуванням поточного шару (рис. 7.4).

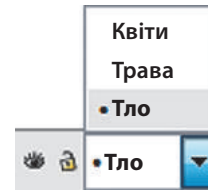


Рис. 7.4

Питання для самоперевірки

1. Для чого об'єднують контури?
2. Як розділити об'єднані контури?
3. Як згрупувати кілька об'єктів?
4. Поясніть різницю між об'єднанням контурів і групуванням об'єктів.
5. Які переваги надає використання шарів?

Вправа 7

1. Побудуйте зображення шляхом групування та використання шарів.

1) Запустіть графічний редактор Inkscape. Побудуйте зображення зірки (рис. 7.5).
2) Згрупуйте об'єкти. Зробіть кілька копій із тлом різного кольору і вишикуйте їх у рядок на однаковій відстані одна від одної.
3) Збережіть зображення у файлі з іменем stars.svg.


Рис. 7.5

2. Створіть багатошарове зображення на одну з тем: «Моє місто», «Садок вишневий коло хати», «Відпочинок біля моря», «Футбольний матч» тощо.

Комп'ютерне тестування

Виконайте тестове завдання 7 із автоматичною перевіркою результату.

Практична робота 2
Створення складених векторних зображень

Завдання: створити бланк грамоти, призначеної для нагородження учасників шкільних спортивних змагань, шляхом копіювання і групування об'єктів різних типів.

Обладнання: комп'ютер зі встановленим графічним редактором Inkscape.

Хід роботи

Під час роботи з комп'ютером дотримуйтеся правил безпеки.

1. Запустіть графічний редактор Inkscape. Оформте грамоту за зразком (рис. 7.6). Для цього побудуйте замкнену криву з трьома вузлами (тло), додайте градієнтне заповнення, налаштувавши його параметри інструментом Градієнт.
2. Додайте написи та побудуйте горизонтальні лінії.
3. Створіть орнамент на основі зірки шляхом копіювання й розмістіть його вертикально.

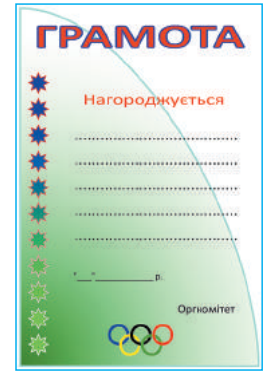


Рис. 7.6

4. Побудуйте емблему і розташуйте внизу.
5. Додайте елементи оформлення на власний розсуд.
6. Збережіть результат роботи у файлі з іменем gramota.svg.

Зробіть висновок: як створити векторне зображення з об'єктів різних типів, як виконати операції копіювання і групування.

РОЗДІЛ 2
КОМП'ЮТЕРНІ ПРЕЗЕНТАЦІЇ

ПОВТОРЮЄМО

У початковій школі ви навчилися працювати з програмою для створення й опрацювання комп'ютерних презентацій — PowerPoint із пакета Microsoft Office (або Impress із пакета Libre Office тощо).

Ви знаєте, що *комп'ютерна презентація* — це електронний документ, який складається з окремих слайдів, призначених для почергового показу на екрані монітора, а за наявності проектора — на великому екрані.

Як вам відомо, створенню презентації передує підготовча робота: продумування мети та сценарію виступу, обмірковування структури презентації тощо. Ви вмієте додавати *слайди*, вставляти на них *тексти*, *малюнки* тощо.

1. Назвіть програми для створення презентацій.
2. Яке призначення комп'ютерної презентації?
3. З чого складається презентація?
4. Назвіть пристрої для демонстрації презентацій.
5. Що передує створенню презентації?
6. Об'єкти яких типів може містити слайд?

У цьому розділі ви навчитеся розробляти презентацію з елементами управління показом, дізнаєтесь, як розробляти дизайн презентації та добирати стильове оформлення слайдів, як додавати до слайдів анімаційні ефекти, звуковий супровід, а також як використовувати гіперпосилання.

§ 8. Системи опрацювання комп'ютерних презентацій

Чи доводилось вам розповідати друзям про літній відпочинок, домашнього улюбленця, своє захоплення? Напевно, ви не обмежувалися усною розповіддю, а показували фотографії, малюнки тощо (рис. 8.1).

Рис. 8.1

Поняття презентації

Згадаємо, що звичайна презентація (від лат. *praesento* — представлення, подання) — це сукупність дій (виступ, лекція, доповідь тощо) і документів (фотографій, плакатів, схем, креслень, аудіо- та відеозаписів, текстів тощо), призначених для донесення до аудиторії певної інформації (про ідею, проект, результат роботи, товар, винахід та ін.).

Сьогодні більшість презентацій замість традиційних засобів наочності: паперових документів, креслень, малюнків, плакатів — використовують можливості сучасної комп'ютерної техніки: цифрові фото, відео, аудіодані та ін.

Комп'ютерна презентація — це набір електронних документів, підготовлених для перегляду на екрані комп'ютера або на великому проекційному чи іншому екрані.

Види презентацій

Залежно від мети презентації доповідач або доповідачка вибирає, як найкраще і найяскравіше донести до аудиторії свою ідею, які можливості сучасних комп'ютерних та інформаційно-комунікаційних технологій доцільно використати.

Види презентацій за способом подання

За способом подання, тобто створення і представлення комп'ютерних презентацій, їх можна розділити на два види: *слайдові* та *потокові* (рис. 8.2).

ВИДИ ПРЕЗЕНТАЦІЙ ЗА СПОСОБОМ ПОДАННЯ		
	Слайдові	**Потокові**
Подання	Послідовність слайдів — окремих екранних сторінок	Відеофільм певної тематики
Тривалість	Тривалість показу кожного слайда може регулюватися доповідачем	Тривалість фільму визначена заздалегідь і може бути змінена призупиненням або прискоренням показу
Особливість	Зручно демонструвати під час виступів, доповідей тощо, де доповідачеві потрібно слідкувати за реакцією аудиторії	Зазвичай демонструються на проекційних або плазмових екранах у торгових і банківських залах, в інших місцях скупчення людей (площі, вокзали тощо)

Рис. 8.2

Обидва види презентацій можуть містити різні об'єкти, наприклад, тексти, графіку, відео, звук тощо.

Види презентацій за призначенням

За призначенням презентації можна розділити на багато видів, оскільки вони використовуються для представлення найрізноманітніших видів людської діяльності (рис. 8.3).

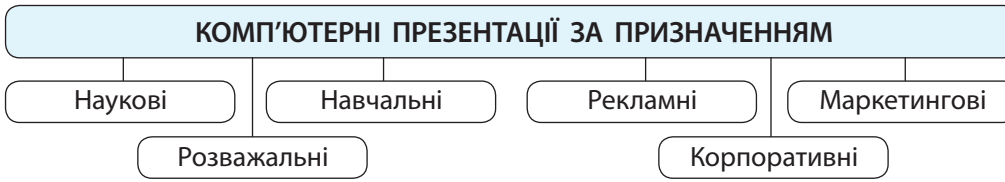


Рис. 8.3

Знайомство із системами опрацювання презентацій

Для створення і представлення комп'ютерних презентацій необхідні певні технічні та програмні засоби.

Прикладне програмне забезпечення для створення й редагування презентацій на комп'ютері називають **системами опрацювання комп'ютерних презентацій**.

Апаратне забезпечення для демонстрації презентацій

Для аудиторного показу апаратне (технічне) забезпечення демонстрації презентацій зазвичай складається із комп'ютера, проектора, екрана та акустичних колонок (рис. 8.4).

Рис. 8.4

Кількі осіб можуть переглянути презентацію на моніторі стаціонарного комп'ютера чи ноутбука, навіть у смартфоні (рис. 8.5).

Рис. 8.5

Програмне забезпечення для опрацювання презентацій

Для створення презентації на комп'ютері необхідно встановити відповідні програми. Нині перелік програмного забезпечення дуже великий. Кожен користувач може вибрати ті програмні засоби, що дозволяють найповніше розкрити задум презентації.

До програмної складової систем опрацювання *слайдових* презентацій відносять PowerPoint (Microsoft Office), Impress (OpenOffice), Apple Keynote, Google Slides, Prezi та ін.
Для опрацювання *потокових* презентацій застосовують Adobe Premiere Elements, Adobe Flash Player, Microsoft Movie Maker та ін.

Тут наведено далеко не повний перелік комерційних і безкоштовних програм для новачків і професіоналів, які призначені для роботи в операційних системах Windows, Mac OS, Linux, Android, iOS тощо та/або з підтримкою веб-опрацювання презентацій.

Знайомство з PowerPoint

З початкової школи вам відомий редактор презентацій Microsoft PowerPoint, який входить до складу комерційного пакета Microsoft Office для Windows і використовується для опрацювання слайдових комп'ютерних презентацій. Доступна і безкоштовна веб-версія цієї програми є у складі пакета програм Office 365.

Як ви знаєте, програма PowerPoint запускається з головного меню Пуск клацанням або значка  , або відповідного ярлика.

Після запуску програми на екрані з'являється вікно, у якому можна вибрати (рис. 8.6) потрібний шаблон з Інтернету, створити нову презентацію або відкрити один з останніх збережених файлів презентації.

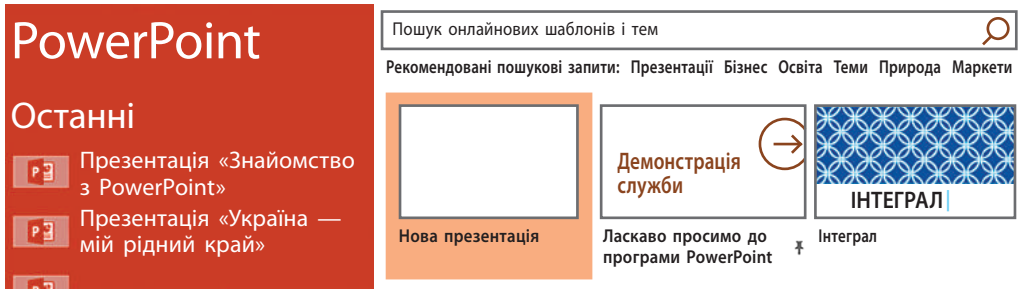


Рис. 8.6

Створення або редагування готової комп'ютерної презентації відбувається в основному вікні програми PowerPoint.

Розглянемо основні об'єкти вікна PowerPoint (рис. 8.7):

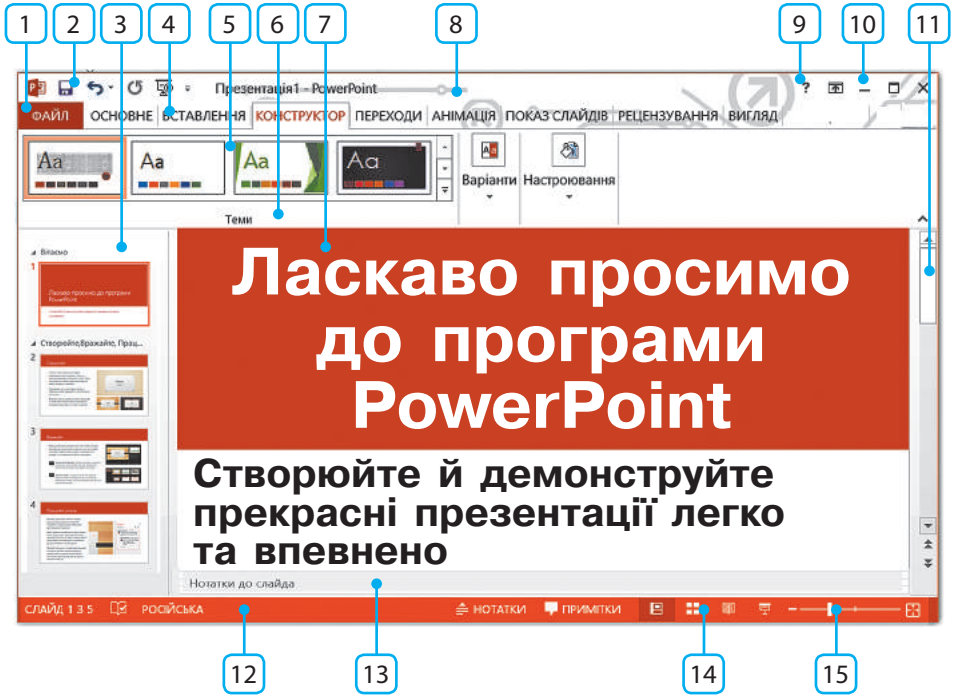


Рис. 8.7

1 — Кнопка Файл (у деяких версіях Офіс) — призначена для створення, відкриття чи збереження файлів презентації та налаштування програми PowerPoint.

2 — Панель швидкого доступу — дублює команди та інструменти, які часто використовуються. Перелік значків можна налаштувати на свій розсуд кнопкою праворуч від панелі.

3 — Область перегляду слайдів (у деяких версіях Слайди) — дозволяє вибирати потрібний слайд, переглядати зміст, змінювати взаємне розташування слайдів, додавати й вилучати слайди тощо. Вигляд налаштовується на вкладці ВИГЛЯД інструментами групи Режим перегляду презентації.

4 — Меню вкладок — містить назви вкладок із набором команд та інструментів. Одинарне клацання назви відкриває відповідну стрічку інструментів, подвійне клацання — згортає (розгортає).

5 — Стрічка інструментів — відображає інструменти активної вкладки.

6 — Групи інструментів — містять інструменти на стрічці, згруповані за певною ознакою, кожна група має свою назву.

7 — Область відображення слайда — відображає макет порожнього слайда або вигляд готового слайда, вибраного в області перегляду, з можливістю вилучення, додавання чи редагування об'єктів на слайді.

8 — Рядок заголовка — відображає назву файла презентації.

9 — Кнопка довідки — відображає вікно автономної або онлайн-довідки.

10 — Кнопки керування вікном — здійснюють згортання, розгортання, закриття тощо.

11 — Смуга прокручування — містить інструменти (бігунок і стрілки) для перегляду тих частин об'єкта, які не поміщаються в області перегляду.

12 — Рядок стану — містить інформацію про зміни в презентації, мову, режим перегляду, масштаб та ін.

13 — Область нотаток — відображає текст нотаток до слайда.

14 — Кнопки режиму перегляду — дублюють інструменти групи Режим перегляду презентації вкладки ВИГЛЯД.

15 — Елементи масштабування — дозволяють змінювати масштаб перегляду слайда в області відображення. Інший спосіб: прокручування коліщатка миші з утримуванням клавіші Ctrl.

Найбільш професійними вважаються презентації Стіва Джобса (рис. 8.8) — засновника всесвітньо відомої корпорації Apple (США). Його виступи з використанням комп'ютерних презентацій захоплювали своєю продуманістю, якістю та простотою.

Рис. 8.8

Навчившись створювати власні презентації, користувачі зможуть зацікавити будь-яку аудиторію своїми проектами, подорожами, захопленнями тощо.

Питання для самоперевірки

1. Чим звичайна презентація відрізняється від комп'ютерної?
2. У чому відмінність слайдової презентації від потокової?
3. Які об'єкти може містити комп'ютерна презентація?
4. Назвіть кілька видів презентацій за напрямком діяльності.
5. Назвіть кілька апаратних складових для аудиторної демонстрації презентації.
6. Яка програма пакета Microsoft Office є редактором презентацій?

Вправа 8

▶▶ Дослідити особливості готової презентації, наприклад однієї з презентацій Стіва Джобса.

1) Запустіть браузер. Знайдіть в Інтернеті відео з будь-якою презентацією С. Джобса, наприклад «Секрети презентацій Стіва Джобса».
2) Перегляньте обране відео або його частину.
3) Запустіть текстовий процесор і створіть новий документ.
4) Запишіть висновки про особливості цієї презентації.
   - Чи читає автор свою доповідь з папірця; зі слайдів?
   - Скільки слів у середньому містить один слайд презентації?
   - Протягом скількох хвилин (секунд) у середньому демонструється один слайд?
5) Збережіть документ із назвою Вправа8 і прокоментуйте висновки.
6) Запустіть програму PowerPoint і створіть нову презентацію. Збережіть файл у форматі PPTX (*тип файла*) із назвою Вправа8 (*ім'я файла*).

Комп'ютерне тестування

Виконайте тестове завдання 8 із автоматичною перевіркою результату.

§ 9. Етапи створення презентації

Підготовка презентацій у PowerPoint тісно пов'язана з опрацюванням текстових документів у MS Word. Разом з тим є особливості, характерні саме для слайдової презентації. Пригадаємо, що комп'ютерна презентація є мультимедійним документом. У ній можна об'єднувати текст, малюнки, звук, відео, анімацію.

Основні етапи розробки презентації

Розробка презентації включає кілька етапів. Розглянемо, які дії потрібно виконати на кожному з них.

1. **Визначення теми й мети презентації.** Чітко формулюємо тему й мету презентації — це дозволить визначити аудиторію, зміст і форму подання матеріалу.
2. **Створення доповіді.** Особливу увагу приділяємо створенню тексту для виступу, обмірковуючи сценарій презентації: що, коли і як демонструватиметься. Іноді текст створюють після розміщення матеріалів на слайдах.
3. **Підготовка та розподіл матеріалу за слайдами.** Визначаємо кількість слайдів із розрахунку 50–70 слів доповіді (до 1 хв) — один слайд. Бажано у тексті вказати номери та/чи заголовки відповідних слайдів.

   Далі відповідно до сценарію презентації підготуємо матеріали: малюнки, відео, звук тощо, обмірковуємо структуру слайдів (далі робота триватиме в середовищі опрацювання презентацій).
4. **Вибір стильового оформлення слайдів.** Залежно від теми та мети презентації використовуємо теми оформлення з колекції PowerPoint, або онлайн-шаблони, знайдені в Інтернеті, або розробляємо власне оформлення.
5. **Розміщення матеріалів на слайдах.** Залежно від виду та обсягу матеріалів вибираємо для кожного слайда готову розмітку (макет) або налаштовуємо власну.

6. **Додавання переходів, анімації, звуку.** Для надання презентації динамічності, акцентування уваги на ключових моментах вмісту застосовуємо звукові та анімаційні ефекти: рух або видозміни об'єктів на екрані, анімовану зміну слайдів (переходи) тощо.
7. **Попередній перегляд презентації.** Переглядаючи презентацію, звертаємо увагу на оформлення слайдів і їх послідовність, на кількість слів і речень на слайдах. Доопрацьовуємо в разі потреби.
8. **Збереження презентації.** Для збереження незавершеної роботи вибираємо формат файла Презентація PowerPoint (*.pptx). Завершену роботу можна зберегти у форматі Демонстрація PowerPoint (*.ppsx). Такі файли відкриваються відразу в режимі перегляду.

Рекомендації щодо оформлення презентацій

Щоб презентація зацікавила аудиторію, варто дотримуватися певних загальних рекомендацій. Ознайомимося з деякими з них.

1. **Час.** Кількість слайдів має приблизно відповідати тривалості виступу у хвилинах (рис. 9.1). Якщо слайдів забагато, доповідач не встигне їх всі показати протягом виступу або доведеться пришвидшити показ. Якщо слайдів замало, це означає, що ви неефективно використовуєте презентацію.

5 слайдів — 5 хвилин

Рис. 9.1

2. **Доповідь.** Гарна доповідь — послідовна, лаконічна й узгоджена з презентацією. Перескакування через кілька слайдів або часте повернення до попередніх слайдів (як і читання тексту зі слайдів) свідчить про непідготовленість доповідача.

3. Вміст слайда. Кожен слайд повинен мати заголовок і якнайменше тексту. Малюнки, фотографії, діаграми варто супроводжувати стислими поясненнями (рис. 9.2).

На слайді вміру тексту — Слайд переобтяжено текстом

Рис. 9.2

4. Розмір слайда (екрана). Залежно від мети презентація може демонструватися на екрані монітора, проектуватися на великий аудиторний екран, друкуватися на папері. Зі списку у вікні Розмір слайда (вкладка Конструктор) можна вибрати один із типових розмірів екрана (слайда) або налаштувати власні, наприклад, для друку на стандартних аркушах паперу.

5. Оформлення слайдів. Для слайдів презентації варто вибирати однотипне оформлення. Можуть відрізнятися стилі слайдів різних розділів, заголовкового та кінцевого слайдів, слайда зі змістом. Для кращого сприйняття тексту читачем варто звертати увагу на шрифт. Потрібно добирати такий шрифт, щоб на екрані (у сантиметрах) він приблизно дорівнював відстані від екрана до читача в метрах (рис. 9.3).

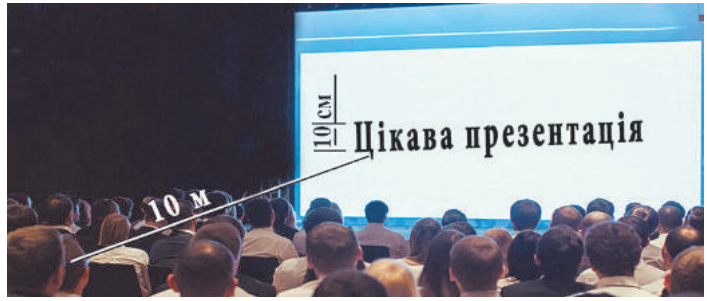


Рис. 9.3

Не варто зловживати кількістю шрифтів: один — для заголовка слайда, інший — для пояснювальних текстів. Також не слід захоплюватися контрастними кольорами тексту і тла. Те, що добре виглядає на моніторі (наприклад, світлий текст на темному тлі), може погано сприйматись на проекційному екрані. Важко розгледіти, скажімо, жовтий текст на білому тлі (рис. 9.4)

Вдала палітра | Невдала палітра

Рис. 9.4

Малюнки, фотографії, діаграми, відео тощо — основа презентації. Проте не варто розташовувати на слайді більш ніж три графічні об'єкти, щоб не розпорошувати увагу глядачів (рис. 9.5).

Слайд добре сприймається | Слайд перевантажено малюнками

Рис. 9.5

▶ 6. **Анімація і переходи.** Анімацію корисно використовувати для пояснення певних динамічних процесів, явищ тощо. Увагу глядачів також привертають переходи, тобто ефектна зміна слайдів (детальніше про це у § 11). Проте ні анімаційними ефектами, ні переходами зловживати не варто.

Дотримання цих рекомендацій не є обов'язковим. Усе залежить від призначення презентації і умов її демонстрації. Так, презентація, використовувана для самоосвіти, може містити набагато більше тексту, ніж презентація, призначена для виступу перед великою аудиторією.

Питання для самоперевірки

1. У чому відмінність презентації від текстового документа?
2. Назвіть основні етапи розробки презентації.
3. У якому форматі доцільно зберігати завершену презентацію для її подальшого перегляду?
4. Скільки, на вашу думку, потрібно слайдів для демонстрації презентації протягом 10 хвилин?
5. Яким приблизно має бути розмір шрифту на слайді, що демонструється на проекційному екрані в аудиторії завдовжки 20 метрів?
6. Поміркуйте, чи є серед наведеного приклади невдалого поєднання кольорів шрифту і тла: а) чорний текст на білому тлі; б) жовтий текст на білому тлі; в) зелений текст на жовтому тлі; г) зелений текст на синьому тлі.
7. Яка кількість зображень є оптимальною для слайда?
8. У яких випадках доречно використовувати анімаційні ефекти?

Вправа 9

Створити й наповнити текстом презентацію на задану тему.

1) Відкрийте створений раніше файл презентації Вправа8 або запустіть PowerPoint і створіть нову презентацію. Кнопкою  на вкладці ОСНОВНЕ створіть *чотири* слайди майбутньої презентації.

2) Придумайте назву презентації (наприклад, «Мій кіт Мурчик») і її вміст (наприклад, ви бажаєте розповісти про те, як Мурчик розважається і що їсть).

3) На *першому* слайді додайте до заголовка назву презентації (наприклад, «Мій кіт Мурчик»). У тексті слайда опишіть у підзаголовку, хто є автором презентації, наприклад «презентація учениці 6–Г класу Сумлінної Поліни».

4) На *другому* слайді додайте до заголовка назву слайда (наприклад, «Розваги Мурчика»). У тексті стисло опишіть розваги Мурчика (наприклад, «Мурчик полюбляє гратися із м'ячиком та клубком ниток»).

5) На *третьому* слайді додайте до заголовка назву слайда, наприклад, «Раціон Мурчика». У тексті коротко опишіть, чим полюбляє ласувати Мурчик (наприклад, ковбаса; сметана).
6) На *четвертому* (завершальному) слайді вмістіть побажання слухацькій аудиторії. Наприклад, Заголовок: «Дякуємо за увагу». Текст слайда: «Я і мій кіт Мурчик бажаємо вам гарного дня». Перегляньте презентацію, відкоригуйте її зміст. Збережіть файл у форматі PPTX із назвою Вправа9.

Комп'ютерне тестування

Виконайте тестове завдання 9 із автоматичною перевіркою результату.

§ 10. Оформлення презентації

Комп'ютерну презентацію і її слайди кожний автор оформлює залежно від теми, змісту, слухачів, власних уподобань тощо.

Шаблони презентацій

Для прискорення роботи над презентацією можна використовувати готові проекти презентацій — так звані шаблони.

Шаблон презентації PowerPoint — це схема (проект) презентації з одного чи кількох слайдів, які оформлені з дотриманням певного задуму.

Зазвичай шаблони PowerPoint зберігаються у файлах з розширенням .potx. Це дає можливість створювати на основі шаблонів готові презентації і зберігати їх в інших форматах без пошкодження вихідного шаблону для повторного використання. Дуже часто як шаблони використовують презентацію у форматі PPTX.

Потрібний шаблон можна відшукати у вікні PowerPoint. Таку можливість надає програма одразу після її запуску (рис. 10.1).

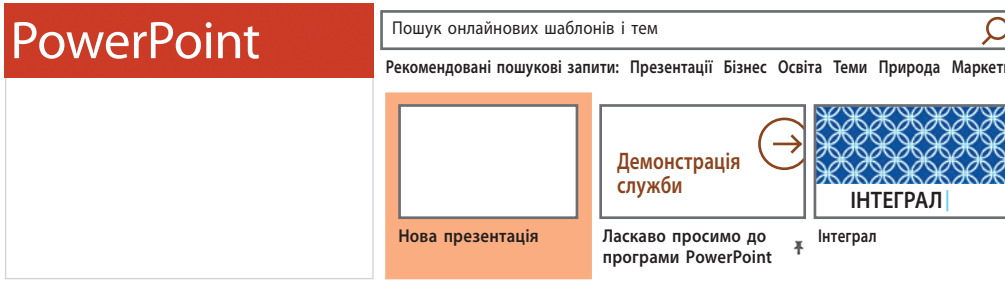


Рис. 10.1

Шаблони і навіть готові презентації різного оформлення та призначення можна безкоштовно завантажити з таких сайтів:

- теми і шаблони Office: templates.office.com
- безкоштовні шаблони для PowerPoint: www.free-power-point-templates.com
- безкоштовні фони для PPT: www.freepptbackgrounds.net
- шаблони PowerPoint: www.indezine.com
- шаблони презентацій PPT: powerpointbase.com; powerpointstore.com
- безкоштовні шаблони PowerPoint: slidehunter.com

Макети слайдів

На слайдах презентації можуть використовуватись різноманітні матеріали, і не завжди вдається швидко дібрати вдалий варіант їхнього взаємного розміщення. Розв'язати цю проблему допоможе використання готових макетів слайдів, вбудованих у PowerPoint.

Макет слайда — це умовна схема слайда, що визначає спосіб розташування об'єктів на слайді.

Щоб **створити новий слайд певного макета**, потрібно:

1) розгорнути список Створити слайд на вкладці ОСНОВНЕ у групі Слайди;
2) клацнути вибраний макет (рис 10.2).

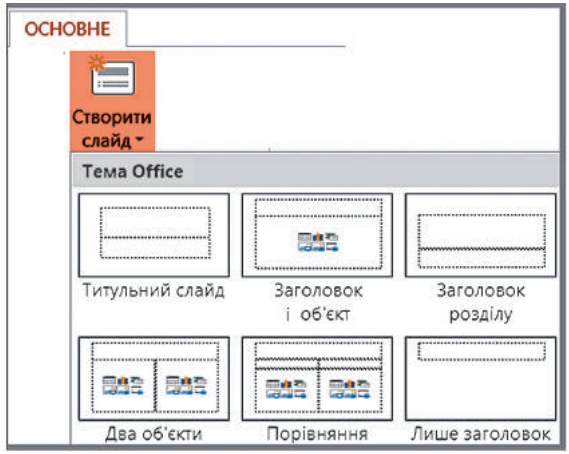


Рис. 10.2

Щоб **змінити макет слайдів**, потрібно:

1) виділити один або декілька слайдів;
2) розкрити на вкладці Основне в групі Слайди список Макет;
3) клацнути один зі зразків макету.

В області відображення слайда з'являться ділянки, обведені прямокутником, так звані *контейнери* (рис. 10.3).

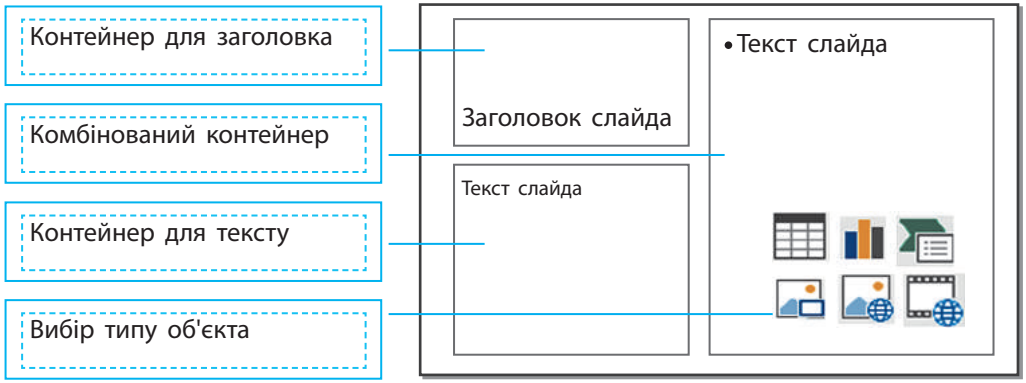


Рис. 10.3

Контейнери різних типів призначені для розміщення в них тих чи інших об'єктів слайда: заголовка, тексту, графічних та інших об'єктів для можливості автоматичного їхнього форматування під час зміни макета та/або оформлення слайда.

Звичайно, на слайд можна додавати власні об'єкти (написи, малюнки тощо). Пам'ятайте, що об'єкти, розміщені поза контейнерами, не форматуватимуться під час автоматичного оформлення слайдів засобами PowerPoint (рис. 10.4).

Порожній макет слайда | Заповнений слайд із доданим написом | Застосування автоматичного шаблону. Доданий поза контейнером напис «загубився»

Рис. 10.4

Кожен тип контейнерів має налаштовані параметри автоматичного форматування вбудованих об'єктів, що дозволяє уникнути невдалих варіантів оформлення слайдів.

Контейнер можна вилучити, клацнувши прямокутну рамку, а потім натиснувши клавішу Delete.

Контейнер для заголовка вилучати небажано, оскільки пізніше, при налаштуванні гіперпосилань (про це — далі), заголовок слайда буде присутній у списку слайдів, що значно полегшує роботу.

Комбінований контейнер дозволяє розміщувати дані різних типів. Для введення тексту достатньо клацнути на вільному місці контейнера і розпочати набирати текст.

Піктограми вибору типу об'єкта розташовані в центрі контейнера, клацання по них дозволяє додати об'єкти різних типів:

Піктограма	Назва	Дія
	Додати таблицю	Відкривається вікно для введення числа рядків і стовпчиків
	Вставлення діаграми	Відкривається вікно Вставка діаграми

Закінчення таблиці

Піктограма	Назва	Дія
	Додавання рисунка SmartArt	Додає графічний об'єкт для наочного подання даних
	Рисунки	Відкривається стандартне вікно вибору файла
	Онлайнові зображення	Відкривається вікно пошуку зображень в Інтернеті
	Вставлення відео	Відкривається вікно вибору відео з локального комп'ютера, з YouTube тощо

Щоб додати новий об'єкт до контейнера, з нього необхідно заздалегідь вилучити об'єкт, який був вставлений раніше.

Теми слайдів

Шаблони презентацій можуть мати різні стилі оформлення слайдів, які називаються темами.

Тема слайда (презентації) — це іменований стиль оформлення об'єктів на слайді: тла, шрифтів, графічних об'єктів тощо.

На вкладці КОНСТРУКТОР доступна стандартна колекція тем оформлення слайдів (рис. 10.5).

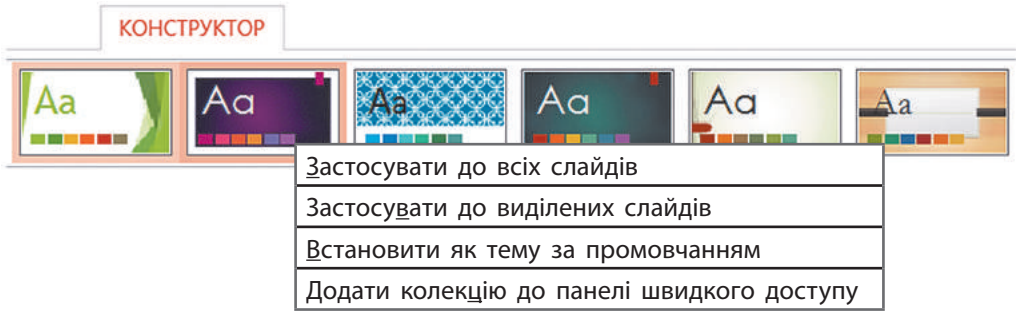


Рис. 10.5

Піктограми в цій колекції ілюструють тло слайда, колірну гаму та стиль тексту кожної з тем оформлення. Достатньо навести вказівник на піктограму, щоб в області відображення слайда одразу побачити відповідне його оформлення.

Якщо ж клацнути одну з піктограм, то всі слайди презентації набудуть обраної теми (стилю) оформлення. Для зміни теми оформлення окремих слайдів слід відкрити контекстне меню обраної теми і вибрати команду Застосувати до виділених слайдів.

Налаштування тем

Кожна тема презентації чи слайда може бути налаштована добором уподобаних варіантів колірної гами слайда, заповнення тла, шрифту тощо. Для цього потрібно на вкладці КОНСТРУКТОР розгорнути список групи Варіанти та обрати потрібне (рис. 10.6)

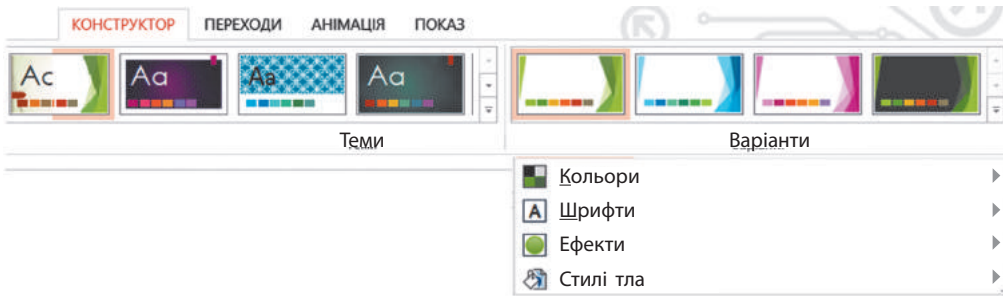


Рис. 10.6

Одна й та сама тема оформлення слайда (наприклад, Ретроспектива) може мати різний вигляд залежно від налаштувань (рис. 10.7).

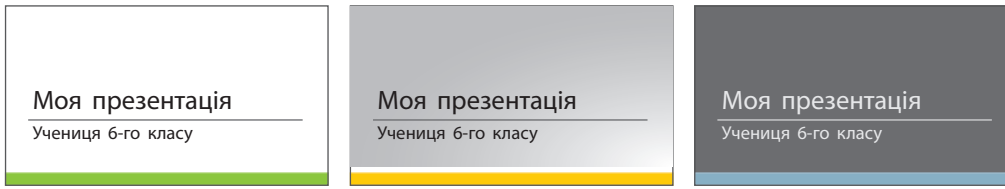


Рис. 10.7

Питання для самоперевірки

1. Що таке шаблон презентації?
2. Де можна знайти і завантажити шаблони презентації?
3. Яким чином можна обрати різні шаблони презентації за допомогою програми PowerPoint?
4. Що визначає макет слайда?
5. Як змінити макет створеного слайда?
6. Яке призначення контейнерів слайда?
7. Чому не варто додавати до слайда об'єкти поза контейнерами?
8. Які об'єкти може містити Комбінований контейнер?
9. Чому небажано вилучати зі слайда Контейнер для заголовка?
10. Що таке тема слайда?

Вправа 10

▶▶ Додати до слайдів графічні об'єкти та оформити презентацію засобами PowerPoint.

1) Відкрийте файл презентації Вправа9. Обміркуйте, до яких слайдів, скільки і які зображення варто додати.
Наприклад, до *другого* слайда (Розваги Мурчика) доцільно додати фото котика, що грається з м'ячиком або клубком ниток, а до *третього* слайда — зображення двох улюблених страв Мурчика (рис. 10.8).

2) Змініть макет *другого* слайда на такий, що містить: Заголовок, один Контейнер для тексту і один Комбінований контейнер (назва макета: Вміст із підписом). У *другому* слайді перенесіть текст із Комбінованого контейнера до Контейнера для тексту. Комбінований контейнер має бути порожнім.

3) До Комбінованого контейнера *другого* слайда додайте фото чи малюнок грайливого котика, скориставшись піктограмами  або  . Під час онлайн-пошуку варто ввести до рядка пошуку ключові слова *котик клубок*.

Рис. 10.8

4) Змініть макет *третього* слайда на такий, що містить Заголовок, два Контейнери для тексту та два Комбіновані контейнери (назва макета: Порівняння). У *третьому* слайді перенесіть тексти із Комбінованого контейнера до двох Контейнерів для тексту. Комбіновані контейнери мають бути порожніми. До кожного з Комбінованих контейнерів *третього* слайда додайте фото чи зображення відповідної страви, скориставшись піктограмами  або  .

5) Застосуйте до *всіх* слайдів презентації одну з тем оформлення, що на вкладці Конструктор. Змініть налаштування кольорових відтінків вибраної теми однаково на *першому* й *останньому* слайдах.

6) Перегляньте презентацію (клавіша F5). За необхідності скоригуйте розмір, колір та розташування текстів і зображень на слайдах презентації (рис. 10.8).
Збережіть файл у форматі PPTX із назвою Вправа10.

Комп'ютерне тестування

Виконайте тестове завдання 10 із автоматичною перевіркою результату.

§ 11. Анімаційні ефекти

Монотонна мова та споглядання статичних зображень і тексту швидко втомлюють слухачів. У багатьох випадках ситуацію можна виправити, якщо додати до об'єктів презентації анімаційні ефекти.

Як вам вже відомо, анімація (з фр. *animation* — оживлення) — це оснащення об'єкта ефектами руху та/або озвучування.

Існує два напрямки використання анімації у презентаціях: *анімація об'єктів на слайдах* і *переходи*, — анімація зміни самих слайдів.

Анімація об'єктів на слайдах

Об'єктом анімації на слайді можуть бути тексти, таблиці, рисунки тощо.

Типи анімацій

Анімаційні ефекти, які можна додати до об'єкта, поділяються на чотири типи: Вхід — задає спосіб появи об'єкта на слайді; Виокремлення — визначає те, як об'єкт поводитиметься впродовж показу слайда; Вихід — задає спосіб зникнення об'єкта зі слайда; Шляхи переміщення — задає швидкість і траєкторію переміщення об'єкта на слайді.

Додавання анімації

До кожного об'єкта можна додати кілька ефектів кожного типу, тим самим програмуючи його досить складну поведінку.

Щоб **додати анімаційний ефект до об'єкта**, потрібно:

1) виділити потрібний об'єкт;
2) розгорнути список ефектів на вкладці АНІМАЦІЯ (група Анімація або список Додати анімацію у групі Додаткові параметри анімації);
3) клацнути піктограму вибраної анімації.

Унизу після всього списку розміщені команди групи Додаткові ефекти для кожного типу анімації. Кожна з цих команд відкриває вікно з великою кількістю додаткових ефектів вибраного

типу. Наприклад, анімаційний ефект зі списку групи Анімація замінює останній доданий ефект, а ефект зі списку Додати анімацію додається поверх вже наявних ефектів.

Кожен анімаційний ефект, доданий до об'єктів, на слайді позначається порядковим номером у квадратику ліворуч від об'єкта в порядку додавання ефектів (рис. 11.1).

Для **вилучення анімаційного ефекту** потрібно клацнути його номер і натиснути клавішу Delete.

Редагування анімації

Для **редагування анімаційного ефекту** потрібно клацнути його номер та скористатись інструментами Параметри ефектів групи Анімація та/або групи Додаткові параметри анімації, та/або групи Хронометраж вкладки АНІМАЦІЯ (рис 11.1).

Таким чином можна налаштувати момент початку ефекту (після клацання мишею; разом із попереднім у списку ефектом; після попереднього), напрямок руху, тривалість анімації тощо.

Анімаційні ефекти зручно редагувати у вікні Область анімації, яке відкривається інструментом із такою самою назвою (рис. 11.1).

Рис. 11.1

Щоб більш детально **налаштувати параметри анімаційного ефекту**, потрібно викликати контекстне меню на його рядку у вікні Область анімації і вибрати команду Параметри ефектів.

Щоб **змінити порядок виконання ефектів**, достатньо всі ефекти перетягти на потрібні місця у списку в цьому ж вікні.

Шляхи переміщення

Як ви вже знаєте, для пояснення динамічних процесів варто створювати на слайді рухомі об'єкти.

Щоб **налаштувати рух об'єкта**, його слід виділити та задати *шлях переміщення* по слайду. Для цього зі списку потрібно вибрати групу Шляхи переміщення, а в ній — інструмент стандартної траєкторії руху або Користувацький шлях, який дозволяє намалювати довільну траєкторію руху (рис. 11.2).

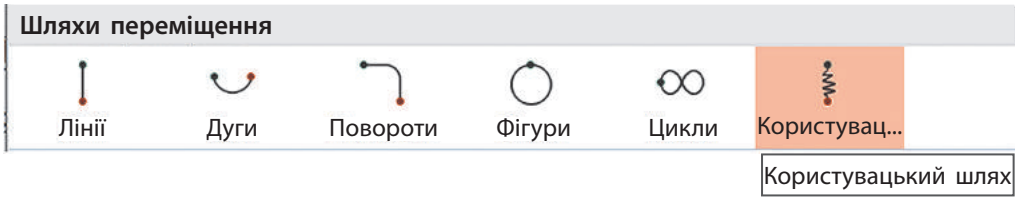


Рис. 11.2

Кінцева точка руху по траєкторії задається користувачем подвійним клацанням миші. Готові шляхи (Лінії, Дуги, Повороти тощо) можна додатково налаштувати перетягуванням круглих маркерів. Вони з'являються, якщо виділити шлях на слайді.

Перегляд анімації

Як у процесі роботи, так і після її завершення результат налаштування анімації можна переглянути в області відображення слайда. Для цього слід натиснути кнопку Відтворити в Області анімації або запустити повноекранний перегляд з поточного слайда.

Переходи

Засоби для встановлення і налаштування ефектної зміни слайдів презентації містяться на вкладці Переходи (рис. 11.3).

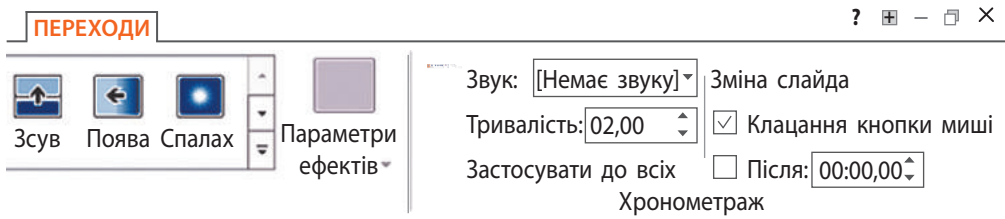


Рис. 11.3

Щоб **додати перехід** до вибраних слайдів, слід клацнути його піктограму в групі Перехід до цього слайда. За замовчуванням перехід додається лише до слайда (слайдів), виділеного в області перегляду.

За допомогою команди Застосувати до всіх (рис. 11.3) вибраний ефект переходу можна застосувати до всіх слайдів презентації.

Налаштування показу слайдів

Інструментами групи Хронометраж можна налаштувати додаткові параметри показу слайдів та переходів: тривалість показу слайда, звук переходу, швидкість переходу, умову переходу тощо.

Для того щоб вкластись у відведений час виступу, можна налаштувати час показу кожного слайда у віконці Після, залишивши можливість переходу клацанням кнопки миші (рис. 11.4).

Перегляд переходу окремого слайда можна здійснити кнопкою Попередній перегляд, розташованою зліва на вкладці ПЕРЕХОДИ у групі Попередній перегляд (рис. 11.5).

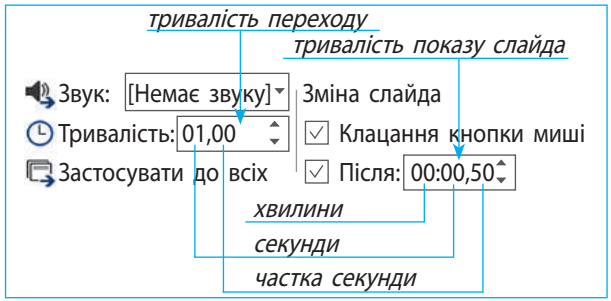


Рис. 11.4

Рис. 11.5

Питання для самоперевірки

1. До яких об'єктів на слайді можна застосувати анімацію?
2. Охарактеризуйте анімаційні ефекти: Вхід, Виокремлення, Вихід, Шляхи переміщення.
3. Опишіть порядок додавання анімаційного ефекту до об'єкта.
4. Як змінити порядок виконання анімаційних ефектів?
5. Як створити для об'єкта довільний шлях переміщення?
6. Як додати до обраного слайда потрібний перехід?

Вправа 11

▶▶ Додати до текстових та графічних об'єктів презентації анімаційні ефекти та налаштувати переходи слайдів.

1) Відкрийте файл презентації Вправа10. Додайте до заголовка *першого* слайда три типи анімації: Вхід, Виокремлення, Вихід. Налаштуйте на *першому* слайді початок усіх анімаційних ефектів Після попереднього.
2) Додайте до рисунка на *другому* слайді анімаційний ефект Шляхи переміщення так, щоб об'єкт почав рухатися довільною траєкторією поза слайдом і закінчив рух на своєму місці. Налаштуйте початок ефекту Шляхи переміщення для рисунка Після попереднього. Налаштуйте тривалість анімаційного ефекту для рисунка — 2 секунди.
3) Додайте до *першого* й *останнього* слайдів однакові переходи. Додайте до *другого* та *третього* слайдів однакові переходи, відмінні від переходів на слайдах 1 і 2.
4) Налаштуйте тривалість *усіх* переходів — 1,5 секунди, а тривалість показу *кожного* слайда — 30 секунд, залишивши можливість зміни слайдів клацанням миші.
5) Додайте до переходу на *останньому* слайді звук Оплески.
6) Перегляньте презентацію та збережіть файл у форматі PPTX із назвою Вправа11.

Комп'ютерне тестування

Виконайте тестове завдання 11 із автоматичною перевіркою результату.

§ 12. Мультимедійний вміст у презентаціях

Презентаційна доповідь стає цікавішою і привабливішою, якщо вона супроводжується аудіо- та відеоматеріалами.

Об'єкти відео

Додавання відеокліпів надає презентації динамічності.

Щоб **вставити відеокліп** у *порожній* Комбінований контейнер слайда, потрібно клацнути піктограму  та у вікні, що з'явилося, вибрати джерело завантаження відеофайла (рис. 12.1).

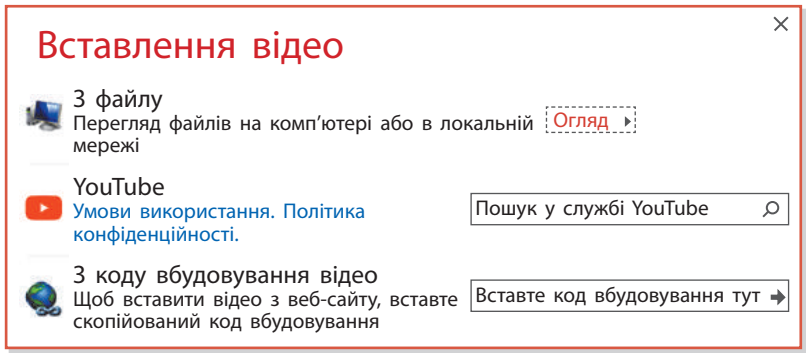


Рис. 12.1

Щоб **вставити відеокліп** у *заповнений* Комбінований контейнер слайда або поза ним, потрібно:

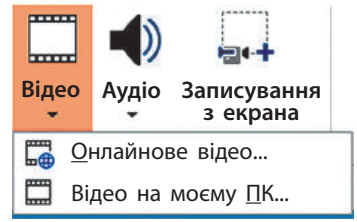


Рис. 12.2

1) на вкладці Вставлення в групі Медіавміст вибрати Відео (рис. 12.2);
2) вибрати джерело, звідки завантажуватиметься відео;
3) вибрати Онлайнове відео (*якщо відео вставляється з Інтернету*), у вікні, що з'явиться, увести назву відео, а після пошуку й вибору потрібного кліпу клацнути кнопку Вставити;
4) вибрати Відео на моєму ПК (*якщо відео розміщене на комп'ютері*), потім у вікні, що з'явиться, знайти потрібний відеофайл та клацнути кнопку Вставити.

Використання відео в презентаціях

Відмінності у відтворенні онлайн-відео та відео з комп'ютера полягає в тому, що комп'ютерний файл убудовується в презентацію і його перегляд не залежить від Інтернету. А онлайн-відео є лише посиланням на файл і за відсутності інтернет-зв'язку не відтворюватиметься.

Форматування (вкладка Формат) рамки, кольорів, відтінків, яскравості тощо стосуються лише відеокліпу, доданого з комп'ютера (або локальної мережі). У випадку завантаження відеокліпу з Інтернету форматування змінює лише статичне зображення титульного кадру.

Під час відтворення всі форматування статичної картинки ігноруються, відеокліп відтворюється у звичайному форматі.

> ! До відеокліпа з Інтернету  не можна застосувати анімаційні ефекти.
>
> До відеокліпа з комп'ютера  анімаційні ефекти застосовуються, як і до інших графічних об'єктів.

Далі будемо розглядати лише налаштування та особливості відеокліпів, які додано з комп'ютера.

Налаштування розмірів відображення та розташування відеокліпа на слайді

Після вставлення відео на слайді з'явиться віконце з його титульним кадром і елементами керування. Як і решту об'єктів (написи, рисунки), розмір відеовікна можна змінювати маркерами, розташованими на його контурі. Перетягуванням за площину віконця його можна розташувати в будь-якому місці слайда (і навіть поза слайдом).

Вилучення непотрібного відео здійснюється за допомогою контекстного меню або клавішею Delete.

Більш детально налаштувати розміри та розташування відеовікна можна в контекстному меню відеооб'єкта. Для цього потрібно скористатися відповідною командою та встановити потрібні значення у вікні налаштувань.

Налаштування перегляду відеокліпа

У режимі редагування під відеокліпом розміщуються елементи керування попереднім переглядом (рис. 12.3).

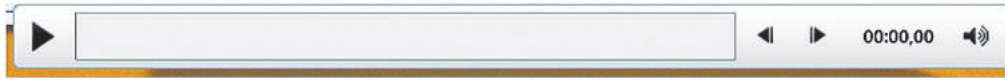


Рис. 12.3

Під час демонстрування презентації відеокліп за замовчуванням містить елементи керування відтворенням кліпу (рис. 12.4).

Рис. 12.4

Особливості відтворення відеокліпа під час презентування можна налаштувати інструментами вкладки ВІДТВОРИТИ. Вони дозволяють не лише налаштувати гучність, тривалість, умови відтворення тощо, а й здійснити обтинання країв відеокліпа за часовими мітками (інструмент Обрізати відеозапис) (рис. 12.5).

Рис. 12.5

Об'єкти аудіо

Для вставлення аудіосупроводу до відеокліпа можна скористатись збереженими на комп'ютері аудіофайлами або за наявності мікрофона зробити власний аудіозапис інструментами Аудіо на моєму ПК… або Записати аудіо… на вкладці Вставлення у групі Медіавміст (рис. 12.6).

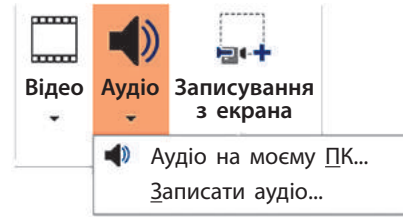


Рис. 12.6

Особливості відтворення аудіофайла можна налаштувати інструментами, що містяться на вкладці ВІДТВОРИТИ.

Кнопка Фонове відтворення вмикає налаштування автоматичного запуску аудіо і відтворення його на всіх слайдах (рис. 12.7).

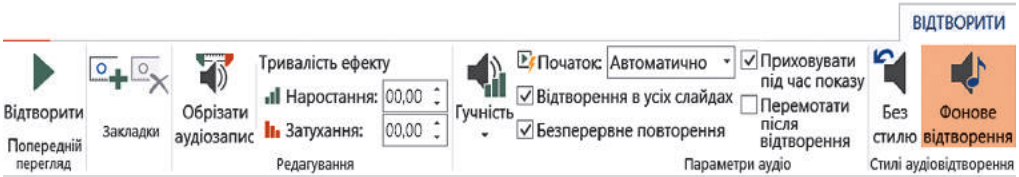


Рис. 12.7

Щоб **записати мовний супровід з мікрофона**, потрібно:

1) вибрати команду Записати аудіо…;
2) у діалоговому вікні, що з'явиться (рис. 12.8), ввести назву звукового фрагмента;
3) клацнути кнопку  для початку записування звука.

Кнопка  призупиняє запис. Для продовження запису, треба знову клацнути кнопку запису.

Кнопка  дозволяє прослуховувати записаний звук.

! Як наслідок, у всіх перелічених випадках на слайді з'явиться піктограма, що зображує гучномовець (рис. 12.9), під яким з'являються елементи керування прослуховуванням, якщо клацнути піктограму чи навести на неї вказівник.

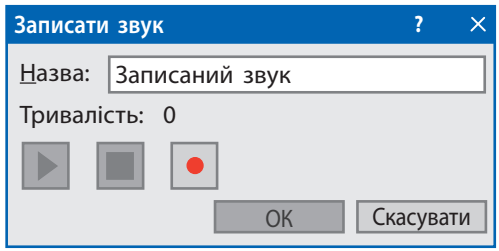


Рис. 12.8

Рис. 12.9

Записування з екрана

PowerPoint 2013 надає можливість здійснити запис відео з екрана (так зване захоплення відео).

Щоб **записати відео з екрана**, потрібно:

1) на вкладці Вставлення у групі Медіавміст вибрати команду Записування з екрана (див. рис. 12.6);
2) у вікні керування, що з'явилося, вибрати тип записуваних даних (аудіо або екранна область).

Якщо обрати Область екрана, то потрібно означити межі області екрана, запис якої вестиметься (рис. 12.10).

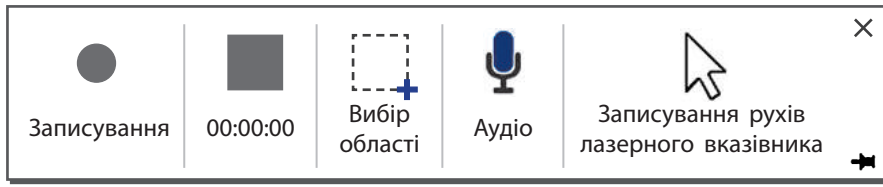


Рис. 12.10

Після початку записування кнопкою  вікно з елементами керування зникає (згортається) і здійснюється запис усього, що відбувається в означеній ділянці екрана. Якщо навести вказівник миші на верхню межу екрана монітора, можна викликати вікно керування записом, або вимкнути автоматичне згортання вікна керування записом, клацнувши на ньому кнопку  (рис. 12.11).

Після завершення записування кнопкою  на слайді з'являється відео з «відзнятим» матеріалом.

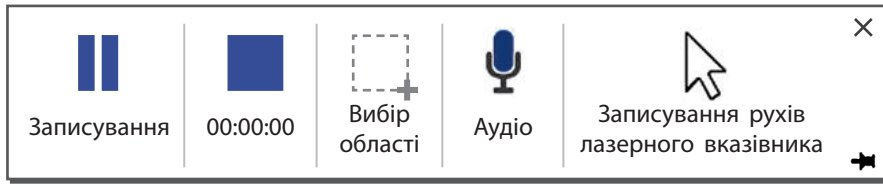


Рис. 12.11

Командою Записування з екрана можна скористатися також і для записування звукового супроводу через мікрофон.

Збереження презентації у відеоформаті

У користувачів нерідко виникає бажання поділитися своєю презентацією, наприклад, у соціальних мережах, або показати товаришеві чи подрузі, у яких на комп'ютері не встановлено програму PowerPoint, переглянути презентацію на відеоплеєрі тощо. У таких випадках є сенс зберегти презентацію у форматі відеофайла.

PowerPoint надає можливість зберігати презентації у відеоформатах MP4 і WMV. Так презентація зі слайдової перетворюється в **потокову**, що являє собою відео, на якому слайди змінюються автоматично через сталий проміжок часу. Усі анімаційні ефекти над об'єктами теж відбуваються автоматично, як і показ вставленого в презентацію відео й увімкнення вставленого аудіо.

Для **збереження презентації в режимі аудіофайла** потрібно:

1) виконати команду Файл → Зберегти як;
2) вказати назву, вибрати місця збереження та тип відеофайла;
3) клацнути кнопку Зберегти та дочекатись завершення збереження, про що свідчить біле індикаторне віконце в рядку стану (рис. 12.12).

Рис. 12.12

До завершення запису не варто редагувати презентацію.

Під час запису не можна закривати вікно програми PowerPoint, оскільки при цьому запис відеофайла буде зупинено.

Питання для самоперевірки

1. Як додати відеокліп до порожнього контейнера слайда?
2. Як додати відеокліп поза контейнером слайда?
3. У чому полягають відмінності онлайнового та комп'ютерного відео, доданих до слайда?
4. Як налаштувати розміри та розташування вікна відеокліпа на слайді?

5. Чи можна відформатувати колірну гаму, яскравість, рамку тощо відеокліпу, вставленого з комп'ютера на слайд?
6. Які інструменти дозволяють налаштувати відтворення відеокліпу під час демонстрації презентації?
7. Яким чином можна обрізати краї відеокліпа засобами програми PowerPoint?
8. Як вставити аудіо з диска комп'ютера на слайд?
9. Яким чином можна створити власний звуковий супровід і додати його до слайда?
10. Як зберегти презентацію у форматі відеофайла?

Вправа 12

Додати до слайдів презентації аудіо- і відеоматеріали та налаштувати їх відтворення.

1) Відкрийте файл презентації Вправа11. Додайте до презентації ще один слайд (на *передостаннє* місце). Додайте заголовок слайда: Відео.
2) Додайте до *нового* слайда будь-яке відео з комп'ютера. Налаштуйте за власним уподобанням розміри і розташування відео. Задайте автоматичне відтворення відео.
3) Відформатуйте вікно відеокліпа, додавши до нього ефектну рамку тощо.
4) Додайте до презентації ще один слайд як *передостанній*. Впишіть заголовок слайда: Записування з екрана. Додайте до цього слайда відеозапис з екрана (наприклад, знайдіть на YouTube уривок з мультфільму про котика).
5) Збережіть файл в одному із відеоформатів. Перегляньте відео.
6) Перегляньте і збережіть файл презентації у форматі PPTX із назвою Вправа12.

Комп'ютерне тестування

Виконайте тестове завдання 12 із автоматичною перевіркою результату.

§ 13. Керування показом презентації

Прості комп'ютерні презентації демонструвати нескладно, існує кілька режимів показу презентацій: звичайний, режим доповідача, довільний показ. Ознайомимося з ними.

Звичайний режим

Вам уже відомо, що повноекранну демонстрацію слайдів презентації у звичайному режимі здійснюють кнопками Із початку (клавіша F5) або Із поточного слайда (сполучення клавіш Shift + F5), що містяться на вкладці ПОКАЗ СЛАЙДІВ.

Якщо під час звичайного режиму показу презентації навести вказівник миші на лівий нижній куток слайда, то з'являться приховані елементи керування показом презентації:

Піктограма	Назва
	Повернення до попереднього слайда (ефекту)
	Перехід до наступного слайда (ефекту)
	Використання вказівника, маркера або пера
	Відображення піктограм слайдів для обрання показуваного
	Масштабування показу окремої частини слайда
	Інші елементи: режим доповідача, режим екрана тощо

Деякі доповідачі демонструють свої комп'ютерні презентації у режимі редагування, щоб мати змогу керувати показом, вибираючи потрібні слайди з Області перегляду слайдів. Проте для аудиторного показу презентацій існує зручний Режим доповідача.

Режим доповідача

Для доповідача важливо не розгубитися під час демонстрування презентації. Тут у пригоді стане можливість додавання прихованих нотаток до слайдів.

Додавання нотаток

Увімкнення й вимкнення відображення області нотаток у режимі редагування презентації здійснюється кнопкою Нотатки (рис. 13.1). Вона міститься на рожевому Рядку стану внизу вікна редактора PowerPoint.

Рис. 13.1

Для створення нотаток доповідачу достатньо в область нотаток увести потрібний текст.

Увімкнення режиму доповідача

Увімкнення показу презентації в режимі доповідача здійснюється сполученням клавіш Alt + F5 (з початку), або Alt + Shift + F5 (з поточного слайда).

Під час показу презентації в режимі доповідача Нотатки відображаються на моніторі доповідача, але не відображаються для слухачів на проекційному екрані.

Таким чином, можна, не засмічуючи слайди текстами, читати текст доповіді з екрана монітора (рис 13.2).

Рис. 13.2

Гіперпосилання

До складних багатослайдових презентацій зручно додавати слайди зі змістом презентації з можливістю автоматичного переходу на потрібний слайд. Організація таких переходів називається посиланням (або гіперпосиланням).

Гіперпосилання (або **посилання**) — це виділений кольором або іншим чином текст, зображення чи кнопка, клацання яких викликає перехід на перегляд інших даних.

Гіперпосилання дають змогу під час перегляду презентації переходити до певних її частин (розділів, слайдів) або на веб-сторінки, відкривати інші документи.

Гіперпосилання можна додати як до текстового, так і до графічного об'єкта.

Щоб **створити гіперпосилання**, потрібно:

1) виділити фрагмент тексту або малюнок;
2) вибрати команду Гіперпосилання (рис. 13.3), що міститься на вкладці ВСТАВЛЕННЯ в групі Посилання;
3) зазначити у вікні, що з'явиться, з чим буде пов'язане гіперпосилання: з файлом або веб-сторінкою; місцем у документі; новим документом; електронною поштою.
4) клацнути кнопку ОК.

Рис. 13.3

Щоб **додати посилання на певний слайд**, потрібно вибрати варіант місцем у документі. У нашому прикладі у вікні (рис. 13.4) для першого пункту слайда ЗМІСТ слід вибрати назву третього слайда, який відкриватиметься після клацання посилання.

Якщо потрібно, щоб після перегляду довільного показу презентація продовжувалася зі слайда, на якому клацнули гіперпосилання (у нас — повернутися до слайда ЗМІСТ), потрібно встановити прапорець Показати та повернутися (рис 13.4).

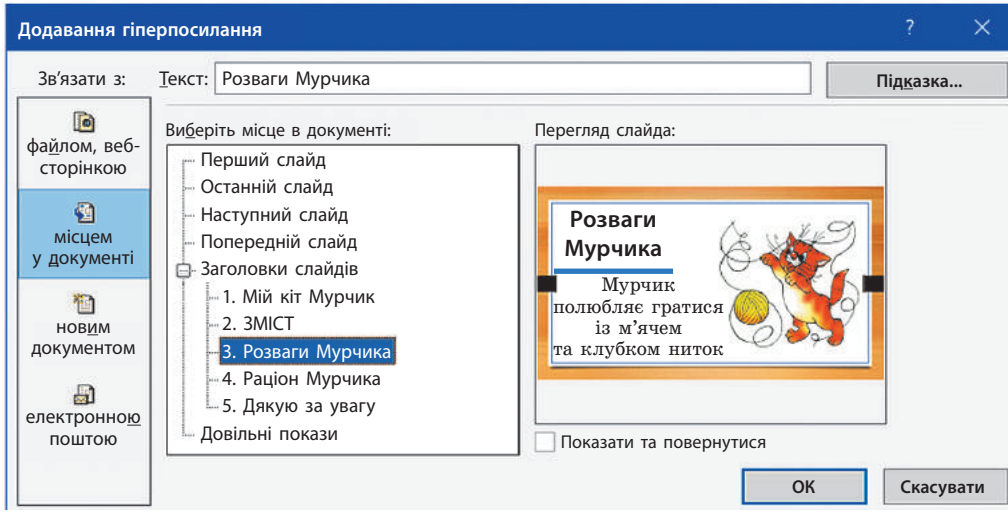


Рис. 13.4

Кнопки дій

Щоб із будь-якого слайда мати можливість повернутися на слайд змісту, можна на цих слайдах створити кнопку дії за алгоритмом: ВСТАВЛЕННЯ → Фігури → Кнопки дії → Кнопка дії: Додому.

Кнопки дій — це низка фігур, що зображують кнопки, для яких вже призначено одну з часто вживаних дій.

Наприклад, для кнопки дій Додому за замовчуванням призначено перехід до першого слайда після клацання миші. Нам же потрібно до цієї кнопки додати гіперпосилання на слайд ЗМІСТ, який заздалегідь було додано до презентації.

Якщо налаштовану певним чином кнопку (або інший об'єкт) скопіювати (Ctrl + C) і вставити (Ctrl + V) до іншого слайда, то зберігаються її розміри, відносне місце розташування на слайді і гіперпосилання. Налаштування, притаманні кнопкам дій, можна застосувати до будь-яких об'єктів слайда: фрагментів тексту, графічних об'єктів тощо.

Довільні покази слайдів

Розглянуті приклади показу презентацій мали або лінійну структуру (при перегляді слайди виводилися на екран послідовно один за одним), або показ слайдів проводився вибірково в ручному режимі чи за допомогою гіперпосилань.

Для налаштування попередньо спланованого вибіркового показу слайдів в автоматичному режимі їх слід згрупувати, створюючи довільні покази.

Довільний показ — це іменована послідовність запланованого показу слайдів презентації.

Довільний показ застосовують, якщо потрібно налаштувати показ слайдів не в тій послідовності чи не в тій кількості, яка є у файлі презентації. При цьому зміст і порядок розташування слайдів у самій презентації залишаються незмінними.

Один і той самий слайд може входити одночасно в кілька *довільних показів*, демонструватися кілька разів в одному *довільному показі* або не демонструватися зовсім.

У презентації щодо захисту проекту з біології можуть бути створені довільні покази (рис. 13.5) «Тварини», «Дерева», «Хвойні», «Листяні» та ін.

Довільний показ додають вже після створення презентації. Слайди, додані пізніше, не відтворюватимуться автоматично.

Щоб **перейти до створення довільного показу**, потрібно:

1) на вкладці Показ слайдів у групі Розпочати показ слайдів натиснути кнопку Настроюваний показ слайдів і вибрати команду Довільний показ;

2) у діалоговому вікні Довільний показ (рис. 13.5) натиснути кнопку Створити;
3) зазначити назву показу в наступному вікні (рис. 13.6), а також сформувати список слайдів у тому порядку, у якому вони демонструватимуться під час довільного показу;

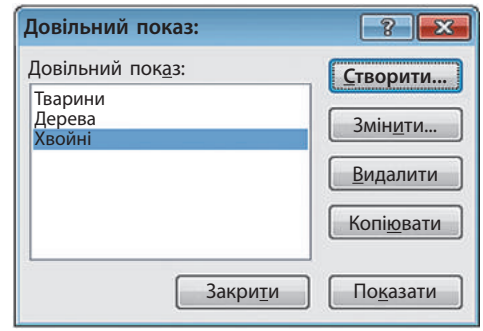


Рис. 13.5

4) додати назву слайда до списку вибраних слайдів кнопкою Додати (рис. 13.6). Під час натискання кнопки Видалити зі списку вилучається назва слайда.

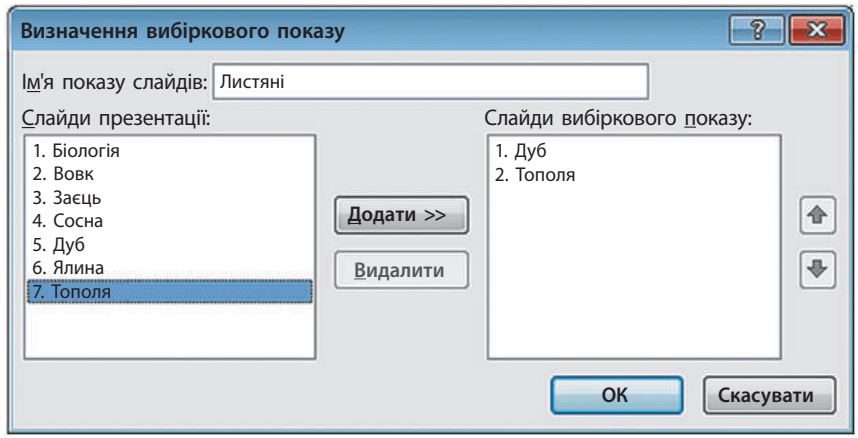


Рис. 13.6

Кнопки зі стрілками, що містяться в ділянці вікна справа (рис. 13.6), дозволяють змінювати порядок слайдів у довільному показі. При цьому порядок слайдів у презентації не змінюється.

Керування показом презентації

Під час планування розробки презентації необхідно урахувати, як буде здійснюватися керування переглядом.

Існують кілька варіантів керування показом презентації:

- *автоматичний* (наприклад, на рекламному екрані на вулиці), при якому зміна слайдів відбувається через певний проміжок часу, заданий під час розробки;
- *керований доповідачем* (повноекранний), за якого доповідач змінює слайди, користуючись пультом дистанційного керування, мишею чи режимом доповідача, зображеним на іншому моніторі;
- *керований користувачем* (віконний), за якого презентація виводиться у вікні, користувач змінює слайди за допомогою смуги прокрутки.

Користувач може вибрати один із варіантів, уточнити додаткові параметри у вікні, що відкривається командою Показ слайдів → Настроювання показу слайдів.

Друк презентації

Попри те що презентація призначається для перегляду на екрані, може виникнути потреба в її роздрукуванні. Список для вибору режиму друку міститься у вікні Друк, яке відкривається командою Файл → Друк.

Залежно від потреби можна роздрукувати:

- *слайди* — друкується по одному слайду на аркуш;
- *видачі* — друкується один слайд (або більше) на аркуш з датою і номером сторінки;
- *нотатки* — друкується один слайд на аркуш і додані до нього нотатки;
- *структуру* — друкується структура презентації у вигляді багаторівневого списку.

Питання для самоперевірки

1. Як запустити звичайну повноекранну демонстрацію презентації з першого слайда; з поточного слайда?
2. Як відобразити приховані елементи керування під час показу презентації?

3. Для чого потрібні нотатки? Як їх додати до слайда?
4. Які переваги режиму доповідача та як його увімкнути?
5. Що таке гіперпосилання?
6. Як додати гіперпосилання на слайді до фрагмента тексту або графічного об'єкта?

Вправа 13

Додати гіперпосилання та налаштувати перегляд презентації.

1) Відкрийте файл презентації Вправа12. Додайте до презентації *другий* слайд з макетом: Заголовок і об'єкт.
2) Додайте до заголовка *другого* слайда слово ЗМІСТ. Додайте до *другого* слайда два рядки тексту, що є заголовками *третього* та *четвертого* слайдів. Наприклад, «Розваги Мурчика» та «Раціон Мурчика». Додайте до кожного рядка тексту на *другому* слайді гіперпосилання на відповідні слайди презентації.
3) Додайте на *третьому* слайді кнопку дії: Додому. Розташуйте кнопку дії у правому нижньому кутку слайда і налаштуйте її розміри. Додайте до кнопки дію на перехід до слайда ЗМІСТ при клацанні мишею. Скопіюйте кнопку дії на *третьому* слайді («Розваги Мурчика») та вставте її до *четвертого* слайда («Раціон Мурчика»).
4) Запишіть нотатки до *третього* й *четвертого* слайдів та перегляньте презентацію в режимі доповідача.
5) Створіть Довільний показ презентації, щоб слайди демонструвались у такому порядку: *перший*, *четвертий*, *третій*, *п'ятий*.
6) Перегляньте презентацію в режимі довільного показу. Збережіть презентацію у форматі PPTX із назвою Вправа13.

Комп'ютерне тестування

Виконайте тестове завдання 13 із автоматичною перевіркою результату.

Практична робота 3
Проектування та розробка власної презентації

Завдання: спроектувати та розробити презентацію за описом.
Обладнання: комп'ютер зі встановленою програмою PowerPoint.

Хід роботи

Під час роботи з комп'ютером дотримуйтеся правил безпеки.

1. Обміркуйте тему п'ятихвилинної презентації, пов'язаної з одним із ваших захоплень.
2. Відкрийте текстовий процесор і надрукуйте текст свого виступу обсягом у 250–300 слів.
3. Окремим кольором позначте в тексті те, якими елементами презентації (фото, малюнки, аудіо, відео тощо) супроводжуватиметься ваш виступ.
4. Обміркуйте, яка кількість слайдів вам потрібна для супроводу виступу (не забудьте додати до основних слайдів три *традиційні*: Титульний, Слайд змісту, Завершальний).
5. Сплануйте, на яких слайдах і які об'єкти розташовуватимуться, проставте в документі номери відповідних слайдів.
6. Підготуйте потрібні графічні, відео- та аудіоматеріали.
7. Запустіть PowerPoint та створіть нову порожню презентацію. Створіть заплановану кількість слайдів із відповідними параметрами макета.
8. Додайте до кожного слайда заголовки і тексти.
9. Додайте до слайдів графічні об'єкти, а за потреби й відео.
10. Налаштуйте загальний дизайн презентації.
11. Налаштуйте переходи слайдів за клацанням миші і часом.
12. Налаштуйте до кількох графічних об'єктів анімаційні ефекти (Вхід, Виокремлення, Вихід) із початком кожного ефекту — Після попереднього. Збережіть файл із розширенням .pptx і назвою Практична3.

Зробіть висновок: для кого призначена створена презентація, у якому режимі її краще демонструвати.

Практична робота 4
Розробка презентації з елементами мультимедіа

Завдання: розробити презентацію з елементами мультимедіа.
Обладнання: комп'ютер зі встановленою програмою PowerPoint.

Хід роботи

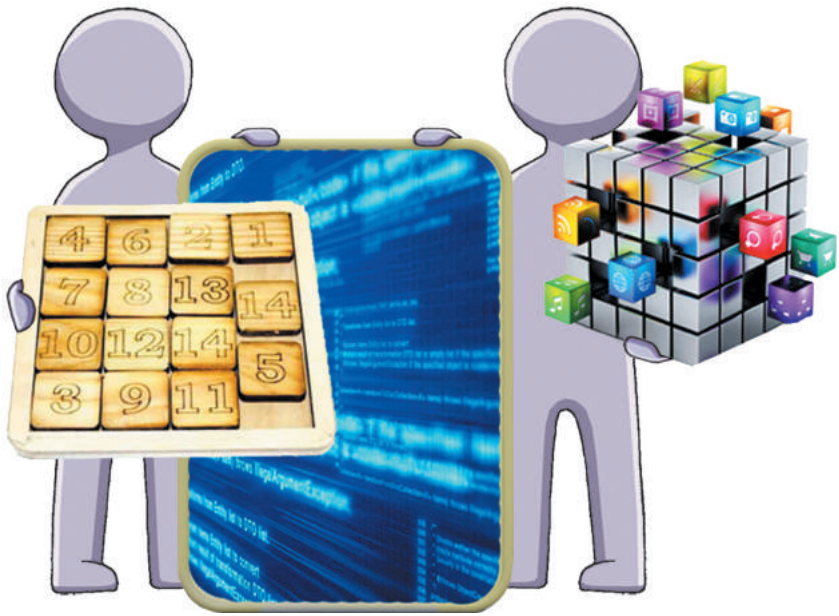
Під час роботи з комп'ютером дотримуйтеся правил безпеки.

1. Відкрийте файл із назвою Практична3 або запропонований вчителем файл презентації з розширенням .pptx.
2. Додайте, якщо такого немає, до презентації слайд із заголовком Зміст.
3. Розмістіть, якщо він в іншому місці, слайд Зміст на другу позицію.
4. Додайте до основного тексту *другого* (*змістового*) слайда заголовки основних слайдів презентації (за винятком *титульного* і *завершального*).
5. На *другому* слайді додайте до кожної назви відповідні гіперпосилання.
6. На *третьому* слайді додайте кнопку дії: Додому.
7. Налаштуйте кнопку дії Додому як перехід до слайда Зміст за клацанням миші.
8. Скопіюйте кнопку дії Додому на решту слайдів (окрім *перших двох*).
9. Додайте до одного зі слайдів (де це доречно) відео.
10. Налаштуйте початок відтворення відео — Автоматично.
11. Перегляньте презентацію, перевірте правильність відтворення відео тощо.
12. Перевірте правильність роботи усіх гіперпосилань. Збережіть файл із розширенням .pptx і назвою Практична4.

Зробіть висновок: чи потрібен Інтернет для демонстрації всіх слайдів створеної презентації, чому кнопку дії Додому не потрібно розміщувати на першому слайді.

РОЗДІЛ 3
АЛГОРИТМИ ТА ПРОГРАМИ

3.1. ОСНОВИ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ



ПОВТОРЮЄМО

Ви знайомі з прийомами роботи в програмному середовищі Python і основами запису алгоритмів мовою Python.

Ви навчилися працювати в інтерактивному режимі, створювати й зберігати програмний код для подальшого використання. Ви вмієте створювати цікаві малюнки за допомогою «черепашачої» графіки.

Для перекладу коду з мови програмування на машинну потрібна спеціальна програма — *транслятор*. Є два основні способи транслювання програми: компіляція й інтерпретація. Під час інтерпретації виконання коду відбувається послідовно, рядок за рядком. У Python використовується саме інтерпретація коду.

1. Які види вікон існують у середовищі Python?
2. Як відкрити вікно програми?
3. Які повідомлення виводяться у вікні консолі?
4. Для чого призначена функція input(); функція print()?
5. Який модуль містить команди черепашачої графіки?

Опановуючи матеріал, ви познайомитеся з принципами об'єктно-орієнтованого програмування, навчитеся створювати мовою Python програми з графічним інтерфейсом, ігрові програми, виконувати обчислення тощо.

§ 14. Класи та об'єкти у програмуванні

Вам відомо, що змінна — це іменована частина оперативної пам'яті, у якій зберігається значення певної величини. Усі величини є об'єктами певного класу. Так, число 7 — це об'єкт, який належить класу (типу) integer. Програмування мовою Python є об'єктно-орієнтованим, тобто програма описує взаємодію об'єктів. Клас задає поведінку об'єктів, створених на його основі. Ми можемо створити власний клас (тип) об'єктів. Уявімо, що потрібно написати програму, в якій моделюється життя акваріумних рибок.

Усі рибки мають такі властивості, як колір, довжина, маса, а також координати положення рибок в акваріумі в певний момент часу. Кожна рибка може рухатися, їсти, гратися з іншими рибками або нападати на них. При цьому рибка росте, змінює положення в акваріумі, тобто змінюються її властивості.

Ми можемо описати клас Рибка, перелічити ті властивості акваріумних рибок, які важливі для нашої програми, і дії, які рибки можуть виконувати. Це допоможе створювати і графічно відображати нових рибок у нашій програмі, програмувати виконання рибками певних дій та зміну значень їхніх властивостей.

Поняття класу

Навколишній світ можна розділити на різні класи речей, такі як «собаки», або «будинки», або «квіти».

Об'єкт — це екземпляр деякого класу.
Клас — це опис об'єктів певного типу.

Об'єкти мають різні властивості. Так, пес Рекс є об'єктом класу Собака. Його властивостями є *порода*, *зріст*, *маса*, *кличка*.

Якщо об'єкт належить до певного класу, то він є екземпляром цього класу. Клас визначає те, яким набором властивостей володіють усі представники класу. Якщо в описі класу Animal (класу Тварини) є властивість *маса*, то вона є в усіх об'єктів цього класу.

Значення властивостей у кожного об'єкта свої.

Створення і використання класу

Створимо клас Dog, що моделює собаку. Що ми знаємо про собак? У них є кличка, вік, маса, порода тощо.

Включимо в клас Dog спільні для собак характеристики: кличка, маса, порода, вік. Ці характеристики стануть атрибутами класу Dog.

Опис класу повідомляє середовищу програмування Python, як створити об'єкт, який представляє модель собаки. Після того як клас буде описано, ми використаємо його для створення об'єктів (екземплярів класу), кожен із яких представлятиме одного конкретного собаку.

Опис класу розташовується на початку коду програми.

Синтаксис опису класу:

```
class Ім'я_класу():
    def ім'я_методу(self, <перелік параметрів>):
        self.змінна = значення
```

1 Опишемо клас Dog().

```
class Dog():
    def __init__ (self, name, breed, age):
        self.name = name
        self.breed = breed
        self.age = age
```

Проаналізуємо структуру опису класу.

- Визначається клас з іменем Dog.
- __init__ — метод, який автоматично виконується при створенні кожного нового екземпляра на базі класу Dog.
- Ім'я методу починається і закінчується двома символами підкреслення.
- Метод __init__ визначається з 4 параметрами в дужках: self, name, breed і age.
- name, breed, age — перелік атрибутів (властивостей) класу.
- У списку всіх параметрів першим має бути параметр self, він потрібен для зв'язку з конкретним об'єктом.

Створення екземпляра класу

Атрибути класу — це імена змінних, у яких зберігаються значення властивостей об'єктів. Описати конкретний об'єкт означає визначити для нього значення атрибутів.

Можна вважати, що клас — це своєрідна інструкція зі створення екземплярів. Відповідно клас Dog — інструкція зі створення екземплярів, які представляють конкретних собак.

Створення об'єкта на основі класу називають створенням екземпляра класу.

Об'єкт створюють за інструкцією:

змінна = Ім'я_класу(<перелік значень атрибутів>)

Далі в програмі з'являється об'єкт, доступ до якого можна отримати через ім'я змінної. Об'єкт отримує атрибути його класу.

Ім'я класу принято починати із символу верхнього регістра (наприклад, Dog), а ім'я окремого екземпляра, створеного на основі класу, записують у нижньому регістрі.

2 Створимо екземпляр класу Dog, який представляє конкретного собаку.

```
my_dog = Dog('Рекс', 'пудель', 5)
```

Створюється екземпляр собаки з кличкою Рекс породи пудель віком 5 років.

Проаналізуємо структуру оператора.

- У процесі оброблення цього рядка Python викликає метод __init__ класу Dog зі значеннями 'Рекс', 'пудель', 5.
- Значення self (посилання на екземпляр) передається автоматично.
- Метод __init__ створює екземпляр, який представляє конкретного собаку, і присвоює атрибутам name, breed, age цього екземпляра передані значення.
  Цей екземпляр зберігається у змінній my_dog.
- При створенні об'єкта класу Dog необхідно надавати значення атрибутів name, breed, age.

! Під час виклику методу дотримуйтесь відповідності між списком значень і списком атрибутів у заголовку методу: кількість; порядок розташування; збіг типів.

```
def __init__(self, name, breed, age):
                    ↑      ↑    ↑
my_dog = Dog('Рекс', 'пудель', 5)
```

Далі ви дізнаєтесь, як використовувати об'єкти в програмі та як змінювати властивості об'єктів.

Питання для самоперевірки

1. Дайте пояснення таких понять у Python: об'єкт, клас, екземпляр класу, атрибут класу.
2. Створіть екземпляр класу Dog з іменем dog1, який представляє 10-річну вівчарку Джека.
3. Опишіть клас Drib, об'єктами якого є звичайні дроби.
4. Створіть екземпляр класу Drib з іменем d1, який представляє значення $\frac{2}{5}$.

Вправа 14

У Василя є S грн. Він хоче пригостити K друзів морозивом, яке коштувало N грн. У магазині з'ясувалося, що вартість морозива збільшилася на P відсотків. Чи вистачить Василю грошей на K порцій морозива? Виконати обчислення для $S = 25$, $K = 5$, $N = 3.5$, $P = 15$.

Виконайте команду ПУСК → Всі програми → Python → IDLE.

1) Запишіть оператор введення з клавіатури значення S — суми грошей, що є у Василя:
```
S = float(input('S = ?'))
```
Натисніть клавішу Enter. IDLE дає підказку S = ? й очікує, поки ви введете дані й натиснете клавішу Enter. Уведіть число 25.

2) Уведіть із клавіатури значення кількості друзів K:
```
K = int(input('K = ?'))
```

3) Уведіть значення N — вартості до подорожчання, P — величини подорожчання у відсотках.
4) Запишіть оператор присвоєння для обчислення NP — нової вартості порції морозива:
```
NP = N+N/100*P
```
5) Запишіть оператор для обчислення KP — кількості порцій, яку можна купити за новою ціною.
6) Запишіть оператор розгалуження для перевірки, чи вистачить для K друзів KP порцій морозива:
```
if (KP > K): print('Так')
else: print('Ні')
```
Отже, чи вистачить Василю грошей на K порцій морозива?

Комп'ютерне тестування

Виконайте тестове завдання 14 із автоматичною перевіркою результату.

§ 15. Властивості об'єктів

Ми створили клас Dog і екземпляр цього класу, який представляє конкретного собаку:

```
my_dog = Dog('Рекс', 'пудель', 5)
```

Розглянемо, як можна дізнатися, які значення мають атрибути об'єкта my_dog, і в разі потреби змінити ці значення.

Доступ до атрибутів об'єкта

Для звернення до атрибутів екземпляра використовується запис через крапку: об'єкт.атрибут. Наприклад, звернення до значення атрибута name екземпляра my_dog:

```
my_dog.name
```

1 Надрукуємо значення атрибута name екземпляра my_dog:
print('Кличка', my_dog.name)

У цьому випадку Python звертається до екземпляра my_dog і шукає атрибут name. Це той атрибут, який позначався self.name у класі Dog. Конструкція my_dog.name отримає значення, що було передане атрибуту self.name при створенні об'єкта my_dog.

2 Виведемо повідомлення про вік собаки:
print('Вік – ', my_dog.age, ' років')
Отримуємо відомості про екземпляр my_dog:
Кличка Рекс
Вік – 5 років
Можна змінити значення атрибута за допомогою оператора присвоєння:
my_dog.name = 'Лессі'

Створення декількох екземплярів

На основі класу можна створити стільки екземплярів, скільки потрібно.

3 Створимо другий екземпляр класу Dog з іменем your_dog:
your_dog = Dog('Альма', 'коллі', 2)
print('Твого собаку звуть', your_dog.name)

У розглянутих прикладах створюються два екземпляри з іменами Рекс і Альма. Кожен із них має той самий набір атрибутів, але їхні значення різні.

4 Об'єкти класу Figura() мають властивості форма і колір. На основі класу Figura створимо два екземпляри класу (рис. 15.1):

```
class Figura():
    def __init__(self, forma, color):
        self.forma = forma
        self.color = color
fg1 = Figura('еліпс', 'red')
fg2 = Figura('трикутник', 'green')
```

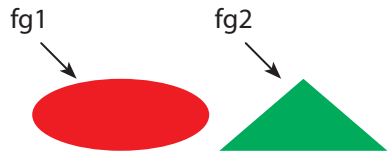


Рис. 15.1

5 Згадаємо черепашачу графіку. У 5 класі ми створювали малюнки з використанням команд модуля turtle. Завантажимо модуль turtle і створимо двох черепашок так само, як ми створювали два екземпляри класу Dog.

Назвемо черепашок Pit і Kate. Виконаємо команди:

```
import turtle
pit = turtle.Pen()
kate = turtle.Pen()
pit.color('blue')
pit.shape('turtle')
kate.color('red')
kate.shape('turtle')
pit.forward(50)
kate.left(90)
kate.forward(100)
```

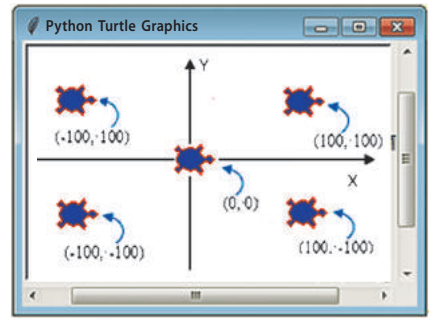


Рис. 15.2

Проаналізуємо програмний код.

- Під час виклику функції turtle.Pen() у середовищі Python створюється об'єкт класу Pen із модуля turtle.
  Якщо збираємося створювати об'єкт, потрібно завантажувати весь модуль (import turtle), а не його окремі методи (from turtle import*).
- Атрибуту shape об'єктів Pit і Kate надається значення turtle.
- Атрибуту color об'єкта Pit надається значення blue, об'єкта Kate — значення red.
- Для об'єкта Pit викликається метод (із ним ми познайомимось пізніше) forward із параметром 50.
- Для об'єкта Kate викликаються методи left та forward.

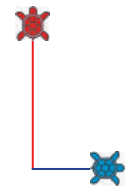


Рис. 15.3

Результат виконання програми наведено на рис. 15.3.

Питання для самоперевірки

1. Як звернутися до значення атрибута певного об'єкта — екземпляра класу?
2. Створіть екземпляр класу Dog з іменем dog2, який представляє дворічного мопса Мулю.

3. Змініть для об'єкта dog2 значення атрибута name.
4. Запишіть оператор для виведення в консоль значень атрибутів об'єкта dog2.
5. Змініть колір об'єкта fg1 (приклад 3) на коричневий.
6. Запишіть оператор для виведення в консоль значення атрибута forma об'єкта fg1 (приклад 3).

Вправа 15

Створити об'єкт класу Тварини.
У Python IDLE виберіть команду File → New File.

1) Створіть клас з іменем Animal(). Метод __init__() повинен містити атрибути, що характеризують вид тварини (kind), зріст (height), кількість лап (legs).

```
class Animal():
    def __init__ (self, kind, height, legs):
            self. kind = kind
            self. height = height
            self. legs = legs
```

2) Уведіть із клавіатури значення характеристик тварини і збережіть значення у змінних k, h, l:

```
k = input('Вид тварини')
h = int(input('Зріст'))
l = int(input('Кількість лап'))
```

3) Створіть екземпляр класу Animal з іменем a1, атрибути якого набувають значення змінних k, h, l:

```
a1 = Animal(k, h, l)
```

4) Виведіть у консоль характеристики об'єкта a1:

```
print( 'Вид – ', a1.kind, 'Зріст – ', , a1.height, 'Лап – ', a1.legs)
```

5) Збережіть файл із назвою Вправа15.
6) Запустіть програму. Уведіть значення для змінних k, h, l, які відповідають об'єкту мавпа.

Комп'ютерне тестування

Виконайте тестове завдання 15 із автоматичною перевіркою результату.

§ 16. Події. Методи

Коли ми складаємо програму, то намагаємось змоделювати певну реальну ситуацію або явище.

Загалом потрібно виконати такі кроки:

1) виявити об'єкти, якими буде маніпулювати наша програма;
2) описати властивості об'єктів;
3) вказати методи (набір дій об'єкта);
4) запрограмувати обробку подій (що об'єкт повинен зробити у відповідь на подію).

Поняття події

У реальному світі події відбуваються навколо нас безперервно. Наприклад, коли собаці Рексу дають команду, він виконує певні дії. У комп'ютерному світі подіями можуть бути натискання кнопки або переміщення миші.

Подія — це вплив на об'єкт, що відбувається в програмі.

Методи — це дії, що можуть виконувати об'єкти даного класу.

Якщо умовно об'єкти вважати іменниками, то їхні методи — дієсловами. Якщо уявити об'єкт як окрему річ, то його методи визначають, як він взаємодіє з іншими речами.

Якщо в програмі для класу Тварини визначити метод їсти, то їсти можуть усі об'єкти цього класу. Визначивши в класі властивості й методи, ми моделюємо об'єкти, властивості які вони мають і дії які можуть здійснювати.

Методи виконуються об'єктом у відповідь на події. Коли з об'єктом Рекс відбулася подія Подано команду "Голос!", він гавкає — виконує дію.

Нехай ми хочемо, щоб при натисканні на певну кнопку малювалася квітка. Для того щоб за допомогою програми намалювати квітку, необхідно дати опис цієї дії, тобто скласти набір покрокових інструкцій, які визначають порядок малювання квітки.

1 Розглянемо взаємозв'язок між властивостями, методами й подіями об'єктів класу Тварини (рис. 16.1).

Об'єкт	Властивість	Метод	Подія
Слон Мавпа Лев	Маса Раціон	Їсти Спати Гарчати	Настала ніч Зголоднів

Рис. 16.1

Опис методів

Методи можуть змінювати значення властивостей (атрибутів) об'єкта, виконувати інші дії над об'єктами.

Синтаксис заголовка методу:

def ім'я_методу(self, <перелік параметрів>):

Замість параметра self у разі виконання методу підставляється ім'я конкретного об'єкта.

Ми вже створили клас Dog. Відомо, що більшість собак вміють сідати й подавати голос за командою. Включимо в клас Dog два види дій (*сідати* й *подавати голос*) (рис. 16.2).

Рис. 16.2

```
class Dog():
    def __init__(self, name, age):
        self.name = name
        self.age = age
    def sit(self):        # Собака сідає за командою
        print(self.name(), ' сідає.')
    def voice(self):    # Собака подає голос за командою
        print(self.name(), ' гавкає. Гав-гав!')
```

У класі Dog ми визначили два методи: sit() і voice(). Поки що методи sit() і voice() обмежуються простим виведенням повідомлення про те, що собака сідає або гавкає. Оскільки цим методам не потрібна додаткова інформація (кличка або вік), вони визначаються з єдиним параметром self. Екземпляри, які будуть створені пізніше, зможуть викликати ці методи.

Виклик методів

Методу необхідно «знати», дані якого об'єкта йому потрібно буде опрацьовувати. Для цього йому як перший аргумент передається ім'я екземпляра класу.

Виклик методу для конкретного об'єкта має вигляд:

об'єкт.ім'я_методу(<перелік значень>)

Створимо екземпляр my_dog на основі класу Dog:

my_dog = Dog('Рекс', 5)

Щоб викликати метод для екземпляра my_dog, укажемо ім'я екземпляра і метод, який викликається, розділивши їх крапкою:

my_dog.sit()

my_dog.voice()

Під час обробки команди my_dog.sit() середовище Python шукає метод sit() у класі Dog і виконує його код. Так само здійснюється обробка команди my_dog.voice().

Об'єкт виконує команди. Отримуємо дії:

Рекс сідає.

Рекс гавкає. Гав-гав!

Робота з класами та екземплярами

Класи використовують для моделювання реальних ситуацій. Можна розробити таку програму, щоб за певним алгоритмом змінювався стан об'єктів.

Розглянемо, як можна змінювати атрибути, пов'язані з конкретним екземпляром класу.

Опишемо клас, що представляє автомобіль.

```
class Car():
    def __init__(self, model, mileage):
        self.mileage = mileage   # Пробіг автомобіля в кілометрах
        self.model = model
my_car = Car('Sens',  2000)
print(my_car.model, ' ', my_car.mileage)
```

(1) (2) (3)

У класі Car визначаємо метод __init__, який набуває параметрів: model, mileage і зберігає в атрибутах екземплярів класу (1).

Створюємо екземпляр класу Car, який зберігається в змінну my_car. При створенні об'єкта вказуємо відповідно до списку атрибутів назву моделі та пробіг (2).

Оператор print виводить у вікно консолі значення атрибутів об'єкта my_car (3): Sens 2000

Ми вже змінювали значення атрибута, звертаючись до екземпляра. Розглянемо приклади.

2 Атрибуту mileage об'єкта my_car надамо значення 120.

```
my_car.mileage = 120
```

До опису можна включити методи, які змінюють деякі атрибути.

3 Додамо до опису класу Car() метод update_mileage() для зміни значення пробігу.

```
class Car():
 ...
    def update_mileage(self, km):  # Встановлення значення пробігу
        self. mileage = km
my_car = Car('Sens', 100)
my_car.update_mileage(200)
print (my_car.model, ' ', my_car.mileage)
```

Буде надруковано: Sens 200.

4 Змінимо код методу update_mileage(), щоб він отримував величину пройденого шляху й додавав її до поточних показників одометра.

```
class Car():
 ...
    def up_mileage(self, km):  # Збільшення значення пробігу
        self.mileage = self.mileage+km
my_car = Car('Sens', 100)
km = 50
my_car.up_mileage(km)
print(my_car.mileage)
```

Буде надруковано: Sens 150.

Змінюючи атрибути, пов'язані з певним екземпляром класу, можна змінювати стан об'єктів у програмі.

Питання для самоперевірки

1. У чому різниця між властивостями й методами об'єкта?
2. Які властивості й методи можуть мати такі об'єкти: учень, учитель, країна, браузер?
3. Поясніть структуру заголовка методу def up_mileage(self, km).
4. Як записати виклик методу def up_mileage(self, km)?
5. Визначте, що надрукує програма:

```
class Car():
    def __init__(self, mileage):
        self. mileage = mileage  # Пробіг автомобіля в кілометрах
    def up_mileage(self, km):
        self.mileage = self.mileage+km
my_car = Car(0)
for x in range(3):
        my_car.up_mileage(50)
        print(my_car.mileage)
```

Підказка:
for x in range(n) :
    <тіло циклу>

Вправа 16

Створити об'єкт класу Тварини.
У Python IDLE виберіть команду File → New File.

1) Створіть клас Animal() з атрибутами kind (вид Тварини) і legs (кількість лап).

```
class Animal():
    def __init__(self, kind, legs):
        self.kind = kind
        self.legs = legs
```

2) Додайте до класу метод description(), який виводить повідомлення про кількість лап в об'єкта — екземпляра класу:

```
def description(self):
    if self.legs == 1: print(self.kind, 'має', self.legs, 'кінцівку')
    else: print(self.kind, 'має', self.legs, 'кінцівки')
```

Підказка:

```
if <Умова> :
    <Оператор 1>
else:
    <Оператор 2>
```

3) Створіть екземпляр класу Animal(), який представляє об'єкт слон:
```
a1 = Animal('слон', 4)
```
4) Запишіть виклик методу description() для об'єкта a1:
```
a1.description()
```

5) Створіть екземпляри класу Animal(), які представляють об'єкти курка, устриця. Запишіть виклики методу description() для цих об'єктів.
6) Збережіть файл із назвою Вправа16. Запустіть програму на виконання. Проаналізуйте результати у вікні консолі.

Комп'ютерне тестування

Виконайте тестове завдання 16 із автоматичною перевіркою результату.

Практична робота 5
Створення програмних об'єктів

Завдання: скласти програму Подорож.
Обладнання: комп'ютер зі встановленим середовищем програмування Python.

Хід роботи

Під час роботи з комп'ютером дотримуйтеся правил безпеки.

У Python IDLE виберіть команду File → New File.

1. Створіть клас з іменем Car().
```
class Car():
    def __init__(self, mileage, fuel):
        self.mileage = mileage    # Пройдений шлях
        self.fuel = fuel          # Кількість пального
```
2. Додайте до класу метод vitr(self), який зменшує значення атрибута fuel:
```
    def vitr(self):     #  Витрати пального на 50 км
```

```
        self.fuel = self.fuel–3
        print('залишилось пального'+str(car1.fuel))
```

3. Додайте до класу метод zapr(self) (заправка), який збільшує значення атрибута fuel на 10 л.

```
    def zapr(self):  # Заправка
        self.fuel = self.fuel+10
        print('залишилось пального', car1.fuel)
```

4. Утворіть екземпляр car1 класу Car() із параметрами 0, 0.

```
car1 = Car(0, 0)
```

5. Виведіть значення пробігу створеного об'єкта.

```
print ('Пробіг', car1.mileage)
```

6. «Заправте» автомобіль: car1.zapr().

7. Запишіть оператор для введення відстані, яку потрібно подолати, і збереження значення у змінній a:

```
a = int(input('відстань?'))
```

8. Поки відстань $a > 0$, потрібно повторювати дії: надрукувати ЇДЕМО!; зменшити значення відстані на 50 км; викликати для об'єкта car1 метод vitr(); якщо пального залишається менше ніж 3 л, викликати для об'єкта car1 метод zapr().

Запишіть оператор циклу while, що реалізує алгоритм руху автомобіля:

```
    while (a>0):
        print ('ЇДЕМО!')
        a = a–50
        car1.vitr()
        if (car1.fuel < 3): car1.zapr()
```

Підказка:
while <Умова>:
    <Оператор>

9. Коли шлях пройдено, повідомте про це:

```
print('ФІНІШ!')
```

10. Збережіть файл з іменем Car.py.

11. Виконайте програму для різних значень відстані, яку потрібно проїхати.

Зробіть висновок: як створювати програмні об'єкти та змінювати їхні атрибути.

§ 17. Створення графічного інтерфейсу

Багато програм використовують графічний інтерфейс, який є більш наочним і зручним для користувача, ніж консоль.

Графічний інтерфейс — це вигляд вікна програми, в якому для взаємодії людини й комп'ютера застосовуються графічні компоненти (вікна, меню, кнопки тощо).

Користувач має довільний доступ (за допомогою клавіатури або миші) до екранних об'єктів — елементів інтерфейсу.

За допомогою мови програмування Python можна створювати програми з графічним інтерфейсом. Для цього в середовищі Python застосовується окремий вбудований модуль tkinter, який містить набір графічних компонентів для створення графічного інтерфейсу. Кожний графічний компонент — це об'єкт певного класу, що має властивості та методи.

Створення вікна програми

Побудова програми починається зі створення вікна. У вікно додаються потрібні компоненти графічного інтерфейсу.

1 Створимо порожнє вікно:

```
from tkinter import *
root = Tk()
root.title ('Графічна програма')
root.geometry('250x150')
root.mainloop()
```

Після збереження програми й запуску на виконання відображається порожнє вікно (рис. 17.1).

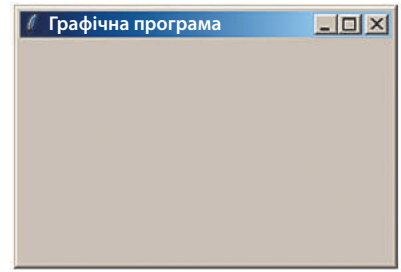


Рис. 17.1

Проаналізуємо наведений код.

- Створюється об'єкт root як новий екземпляр класу Tk(), який визначений у модулі tkinter. Через змінну root ми можемо керувати атрибутами вікна.

- За допомогою методу title() додають заголовок вікна.
- За допомогою методу geometry() встановлюють розміри вікна (у пікселях). Як параметр методу geometry() передається рядок у форматі «ширина x висота». Якщо під час створення вікна програми метод geometry() не викликається, то вікно займає простір, необхідний для розміщення внутрішнього вмісту.
- Метод mainloop запускає головний цикл обробки подій.

Рядок root.mainloop() має бути останнім рядком програмного коду.

Початкова позиція вікна

За замовчуванням вікно позиціонується у верхній лівий кут екрана. Але ми можемо змінити його положення, додавши потрібні значення в метод geometry():

```
root.geometry('250x150+300+250')
```

Тепер рядок у методі geometry має такий вигляд: ширина x висота + координата X + координата Y. Тобто при запуску вікно розміщуватиметься на 300 пікселів праворуч і на 250 пікселів униз від верхнього лівого кута екрана.

Додавання віджета

Користувач в основному взаємодіє з програмою за допомогою різних кнопок, меню, значків, уводячи інформацію в спеціальні поля, вибираючи певні значення у списках тощо. Ці графічні компоненти називатимемо віджетами (від англ. *widget*).

Віджети — це блоки для створення графічного інтерфейсу програми, тобто будівельні цеглинки програми.

Більшість віджетів є стандартними в усіх візуальних мовах програмування.

Модуль tkinter містить описи класів графічних компонентів, і ми будемо створювати віджети як екземпляри цих класів.

Віджет Label містить рядок (або декілька рядків) тексту й застосовується для виведення заголовків, підписів елементів інтерфейсу тощо.

Об'єкт Label створюється викликом класу Label модуля tkinter. При цьому обов'язковим аргументом є лише батьківський віджет (всередині якого створюється напис). Інші властивості можуть задаватися (змінюватися) пізніше.

Синтаксис створення віджета:

змінна = Label(батьківський_віджет, [властивість = значення])

> ! Квадратні дужки в переліку аргументів вказують на те, що ці параметри задавати необов'язково.

При створенні об'єкта Label можно задати такі властивості:

- text — текстовий рядок;
- width, height — ширина і висота у знакомісцях (кількість символів);
- bg, fg — колір фону і символів;
- font — параметри шрифту (тип, кегль).

Наприклад, font = 'Arial 18' — шрифт Arial, 18 кегль.

Створений об'єкт потрібно відобразити у вікні. Найпростіший спосіб — це використання методу pack:

змінна.pack()

Якщо не вставити цей рядок, то напис у вікні так і не з'явиться, хоча він є в програмі.

2 Додамо у вікно напис Hello, World! (рис. 17.2).

Рис. 17.2

```
from tkinter import*
root = Tk()
root.title('Привітання')
root.geometry('300x100')
lab = Label(root, text = 'Hello, World!', font = 'Arial 18', bg = 'blue',
fg = 'yellow')
lab.pack()
root.mainloop()
```

Напис з'явиться у верхньому лівому куті вікна.

Розташувати об'єкт в іншій частині вікна можна за допомогою метода place(x, y), де параметри x і y установлюють зміщення елемента по горизонталі та вертикалі відносно верхнього лівого кута вікна:

змінна.place(x = <значення>, y = <значення>)

3 Застосуємо метод place() замість методу pack().

Замінимо оператор lab.pack() оператором lab.place(x = 80, y = 30)

Тепер верхній лівий кут напису зміщено на 80 пікселів від лівого краю вікна і на 30 пікселів від верхнього краю (рис. 17.3).

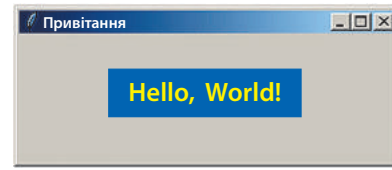


Рис. 17.3

Зміна властивостей віджета

У ході роботи з програмою вже створені об'єкти можуть змінювати властивості. Наприклад, нам може знадобитися змінити текст напису або колір і розмір символів. Змінити значення атрибутів віджета можна за допомогою методу config.

Синтаксис виклику методу:

об'єкт.config (властивість = значення)

4 Змінимо для раніше створеного об'єкта lab текст напису, колір тла й символів і помістимо напис у лівий верхній кут вікна (рис. 17.4).

Для цього потрібно додати до програмного коду (але перед рядком root.mainloop()!) оператори:

```
lab.config(text = 'Я вивчаю Python', bg = 'grey', fg = 'white')
lab.place(x = 1, y = 1)
```

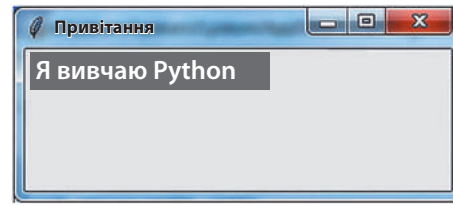


Рис. 17.4

Отже, під час розробки програм із графічним інтерфейсом для того, щоб розмістити у вікні програми графічні компоненти, потрібно створити об'єкт і задати значення атрибутів (налаштувати його властивості).

Питання для самоперевірки

1. Опишіть хід побудови графічного інтерфейсу програми.
2. Опишіть порядок дій під час створення вікна програми.
3. Як створити об'єкт класу Label?
4. Як змінити властивості вже створеного віджета класу Label?
5. Яким оператором має завершуватися код програми з графічним інтерфейсом?

Вправа 17

Створити об'єкт класу Label.

У Python IDLE виберіть команду File → New File.

1) Створіть вікно програми розмірами 300 × 300 із заголовком Перша програма.

```
from tkinter import*
root = Tk()
root.title('Перша програма')
root.geometry('300x300')
```

2) Додайте до вікна віджет lab1 класу Label. Задайте такі значення атрибутів: текст напису: Привіт, друже!; параметри шрифту: Arial, 18 кегль; колір тла — жовтий ('yellow'); колір символів — сірий ('grey'); ширина — 20 знакомісць.

```
lab1 = Label(root, text = 'Привіт, друже!', width=20, font = 'Arial 18',
bg = 'yellow', fg = 'grey')
lab1.pack()
```

3) Заверште програму оператором запуску циклу обробки подій: root.mainloop()
4) Збережіть код з іменем Вправа17 і виконайте програму.
5) Змініть для об'єкта lab1 текст напису на Я програмую на Python, колір тла та літер на свій розсуд і перемістіть напис у центр вікна.
6) Збережіть програмний код і виконайте програму.

Комп'ютерне тестування

Виконайте тестове завдання 17 із автоматичною перевіркою результату.

§ 18. Опрацювання подій

Ви маєте досвід роботи з різними програмами і, звісно, погодитесь, що програма повинна мати зручний інтерфейс, який передбачає діалог програми з користувачем.

З'ясуємо, як «примусити» програму реагувати на дії користувача.

Клас Button

Командна кнопка — один із найрозповсюдженіших компонентів у програмах із графічним інтерфейсом, що застосовується для запуску чи закінчення певного процесу.

Синтаксис створення об'єкта класу Button:

змінна = Button(батьківський_віджет, [властивість = значення])

Під час створення об'єкта Button можна задати властивості для налаштування вигляду кнопки, які ми розглянули в ході знайомства з об'єктами класу Label: text, width, height, bg, fg, font.

1 Створимо командну кнопку із написом ok (рис. 18.1).

```
from tkinter import*
root = Tk()
root.geometry('200x100')
btn = Button(root, text = 'ok', width = 10,
height = 2, font = 'Arial 16')
btn.pack()
root.mainloop()
```

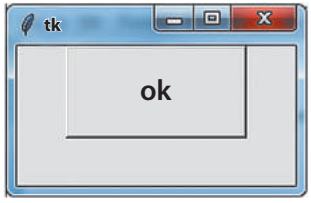


Рис. 18.1

Під час створення навчальних та ігрових програм буває доречно додавати на кнопку не напис, а малюнок. Для цього потрібно задати значення атрибута image.

Якщо малюнок зберігається в окремому файлі, то потрібно зазначити його місцерозташування за допомогою функції PhotoImage(file = 'ім'я файла').

Засобами tkinter можна завантажувати тільки зображення формату GIF (файли з розширенням .gif).

2 Створимо командну кнопку і додамо на кнопку малюнок із файла Picture.gif, який міститься в тій самій папці, що і файл програми.

```
from tkinter import*
root = Tk()
my_image = PhotoImage(file = 'Picture.gif')
btn = Button(root, image = my_image)
btn.pack()
root.mainloop()
```

Якщо виконати цей код, побачимо вікно програми (рис. 18.2). Наша програма виконується — створено вікно з командною кнопкою, на якій міститься малюнок. Проте після клацання кнопки не відбувається жодних дій, бо не створено метод — обробник події Натискання на кнопку.

Рис. 18.2

Обробка події Натискання на кнопку

Основна риса програм із графічним інтерфейсом — *інтерактивність*. Програма не просто виконує оператори коду від початку до кінця, а залежать від дій користувача. Така програма очікує від елементів інтерфейсу подій і виконує у відповідь на події методи — прикріплені до цих віджетів обробники подій.

Щоб прикріпити до віджета обробник події Натискання на кнопку, необхідно під час створення цього об'єкта в переліку атрибутів указати параметр command і присвоїти йому посилання на метод, який буде виконуватися в разі натискання (рис. 18.3).

```
def btn_click():
...
btn = Button(root, text = 'ok', command = btn_click)
```

Рис. 18.3

3 Додамо обробник події до командної кнопки btn:

```
from tkinter import*
def btn_click():
    root['bg'] = 'green'
root = Tk()
root.geometry('200x100')
btn = Button(root, text = 'ok', command = btn_click)
btn.pack()
root.mainloop()
```

Тепер програма реагує на натискання кнопки — фон головного вікна змінюється на зелений.

> **!** Синтаксис опису обробника події такий самий, як у методу класу, але в дужках відсутній параметр self.

4 Створимо «стрибаючу» кнопку (рис. 18.4). Після натискання на кнопку координати розташування кнопки змінюються випадковим чином.

```
from random import randint
from tkinter import*
def btn_click():
    x1 = randint(5, 190)
    y1 = randint(5, 95)
    btn.place(x = x1, y = y1)
root = Tk()
root.geometry('200x100')
btn = Button(root, text = 'OK', command = btn_click)
btn.place(x = 10, y = 10)
root.mainloop()
```

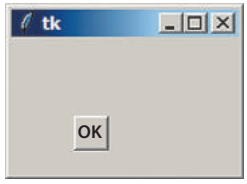


Рис. 18.4

Отже, функція btn_click() виконується після кожного натискання кнопки, і об'єкт btn змінить своє положення у вікні.

Як бачимо, кнопки є дуже популярними й корисними елементами графічного інтерфейсу.

Для організації діалогу користувача з програмою нам також потрібно мати можливість вводити дані. З цією можливістю ми ознайомимося далі.

Питання для самоперевірки

1. Опишіть послідовність дій під час створення командної кнопки.
2. Які атрибути відповідають за зовнішній вигляд кнопки?
3. Як додати до об'єкта Button обробник події Натискання на кнопку?
4. Як додати на кнопку малюнок?
5. Назвіть оператор, яким потрібно скористатися для зміненя кольору форми на червоний.
6. Визначте, як виглядає кнопка, створена з такими значеннями атрибутів:
   btn = Button(root, text = 'Натисни мене', width = 20, height = 2, bg = 'blue', fg = 'white')

Вправа 18

Створити програму з кнопками і написом.

У Python IDLE виберіть команду File → New File.

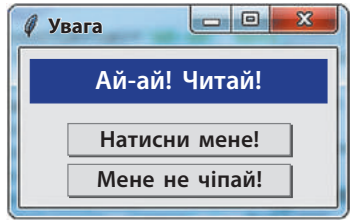


Рис. 18.5

1) Завантажте модуль tkinter і створіть вікно програми розмірами 220 × 110 із заголовком Увага (рис. 18.5).
2) Додайте до вікна віджет lab класу Label:
   lab = Label(root, text = ' ', font = 'Arial 16', width = 15, bg = 'blue', fg = 'white')
   lab.place(x = 20, y = 10)
3) Додайте до вікна віджет bA класу Button:
   bA = Button(root, text = 'Натисни мене!', command = bA_click)
   bA.place(x = 30, y = 50)
   Додайте до вікна віджет bB класу Button із заголовком Мене не чіпай!
4) Після рядка from tkinter import* запишіть обробники події Натискання на кнопку для кожного з об'єктів bA і bB:

```
def bA_click():
    lab.config(text = 'Молодець!')
def bB_click():
    lab.config(text = 'Ай-ай! Читай!')
```

5) У кінці програмного коду напишіть оператор:
root.mainloop()
6) Збережіть програмний код з іменем Вправа18 і виконайте програму.

Комп'ютерне тестування

Виконайте тестове завдання 18 із автоматичною перевіркою результату.

§ 19. Організація діалогу з користувачем

Віджет Label дозволяє виводити інформацію, наприклад результати обчислень, текстові повідомлення. Але можливість вводити дані повинен мати й користувач. Таку можливість надає використання віджетів класу Entry.

Клас Entry

Компонент Entry — це поле для введення тексту, також його можна використовувати і для виведення.

Синтаксис створення об'єкта класу Entry:

змінна = Entry(батьківський_віджет, [властивість = значення])

Для налаштування вигляду віджета Entry слід задати значення атрибутів, спільних із віджетами класів Button і Label: text, width, height, bg, fg, font.

Головним атрибутом віджета Entry є атрибут text, який прив'язує віджет до текстової змінної, в якій зберігатиметься інформація для введення/виведення через поле віджета Entry.

1 Створимо об'єкт entry1 із такими атрибутами:
- текст, уведений у текстове поле, зберігається у змінній *s*;
- ширина текстового поля — 14 знакомісць;
- параметри шрифту — Arial, 18 кегль.

```
entry1 = Entry(root, text = s, width = 14, font = 'Arial 18')
```

Методи Entry

Розглянемо основні методи елемента Entry.
- Метод get() дозволяє отримати значення, що міститься в текстовому полі.

2 Якщо назвати об'єкт класу Entry іменем entry1, то отримати значення і присвоїти його змінній *a* цілого типу можна так:

```
a = int(entry1.get())
```

- Метод insert(index, str) виводить у текстове поле рядок, починаючи зі знакомісця з номером index.

3 Виведемо до поля об'єкта класу Entry значення змінної *x*:

```
entry1.insert(0, x)
```

- Метод delete(first, last) вилучає символи, починаючи зі знакомісця з номером first, до знакомісця з номером last (нумерація символів починається із 0). Щоб вилучити весь текст, як другий параметр потрібно указати END. Бажано перед викликом методу insert() записувати виклик методу delete(), тобто перед виведенням очищувати текстове поле.

4 Очистимо текстове поле перед виведенням:

```
entry1.delete(0, END)
entry1.insert(0, 'Молодець!')
```

5 Видалимо символи з 7-го по 12-й із текстового поля (рис. 19.1):

```
from tkinter import*
def b1_click():
    entry1.delete(7, 12)
root = Tk()
```

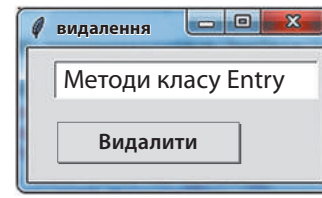


Рис. 19.1

```
root.title('видалення')
root.geometry('220x100')
s = 'Методи класу Entry'
entry1 = Entry(root, text = s)
entry1.place(x = 20, y = 10)
entry1.insert(0, s)
b1 = Button(root, text = 'Видалити',
command = b1_click)
b1.place(x = 20, y = 50)
root.mainloop()
```

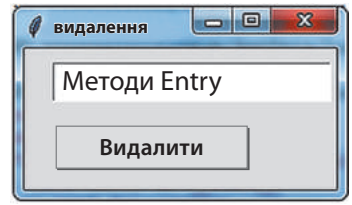


Рис. 19.2

Текст у полі об'єкта entry1 змінюється після натискання кнопки Видалити (рис. 19.2).

6 Уведемо число до поля об'єкта entry1 і виведемо квадрат цього числа до поля об'єкта entry2 (рис. 19.3).

```
from tkinter import*
def b1_click():
    a = float(entry1.get())
    b = a ** 2
    entry2.delete(0, END)
    entry2.insert(0, str(b))
root = Tk()
root.title('Квадрат числа')
root.geometry('220x160')
laba = Label(root, text = 'a =', font = 'Arial 18')
laba.place(x = 10, y = 10)
s = ''
entry1 = Entry(root, text = s, width = 8, font = 'Arial 18')
entry1.place(x = 80, y = 10)
labb = Label(root, text = 'a ** 2 =', font = 'Arial 18')
labb.place(x = 10, y = 110)
entry2 = Entry(root, text = s, width = 8, font = 'Arial 18')
entry2.place(x = 80, y = 110)
b1 = Button(root, text = 'Обчислити', command = b1_click)
b1.place(x = 20, y = 50)
root.mainloop()
```

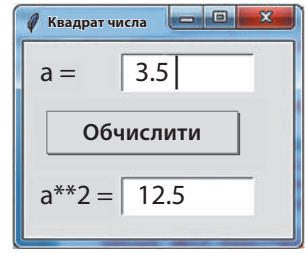


Рис. 19.3

Вікно повідомлень

Іноді буває необхідно вивести певне повідомлення про роботу програми в діалоговому вікні, не створюючи спеціального віджета. Пакет tkinter містить модуль messagebox, який надає доступ до вікон повідомлень.

Модуль messagebox потрібно імпортувати додатково:

```
from tkinter import messagebox
```

Для того щоб згенерувати вікно повідомлення з кнопкою Ок, потрібно для об'єкта messagebox викликати методи showerror(), showinfo() або showwarning(). Від вибраного методу залежить вигляд піктограми у вікні повідомлення (рис. 19.4).

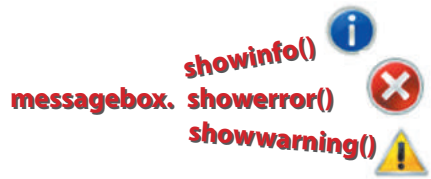


Рис. 19.4

Синтаксис виклику методу такий:

```
messagebox.showinfo(заголовок, текст)
```

7 Згенеруємо вікно повідомлення інформаційного характеру (рис. 19.5):

```
messagebox.
showinfo('Інформація',
'Обчислення завершене')
```

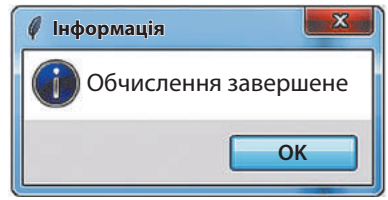


Рис. 19.5

8 Напишемо обробник події Натискання на кнопку, в якому з полів об'єктів entry1 і entry2 класу Entry зчитуються значення для змінних a і b; якщо $b = 0$ (рис. 19.6), генерується вікно з повідомленням про помилку (рис. 19.7), інакше генерується вікно зі значенням a / b.

```
def b1_click():
    a = float(entry1.get())
    b = float(entry2.get())
    if b == 0:
        messagebox.showerror('Помилка', 'На 0 ділити не можна!')
    else:
        messagebox.showinfo('Відповідь', str(a/b))
```

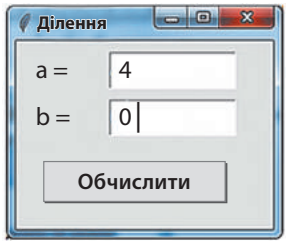


Рис. 19.6

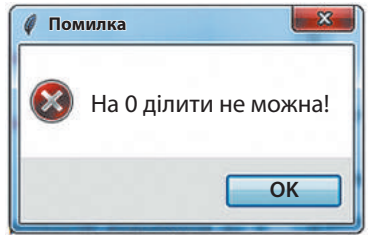


Рис. 19.7

Питання для самоперевірки

1. Поясніть синтаксис створення об'єкта класу Entry.
2. Яким атрибутам слід задати значення для налаштування вигляду віджета Entry?
3. Опишіть основні методи об'єкта класу Entry.
4. У поле об'єкта entry класу Entry введено рядок Інформатика. Який рядок міститься в полі після виконання оператора: а) entry1.delete(2, 6); б) entry1.insert(3, '111')?
5. Який вигляд має вікно повідомлення, що створене таким оператором: messagebox.showinfo('Підказка', 'Уведіть число')?

Вправа 19

▶▶ Створити програму для обчислення значення виразу. У Python IDLE виберіть команду File → New File.

1) Завантажте модуль tkinter і створіть вікно програми розмірами 250 × 200 із заголовком Розв'язувач (рис. 19.8).

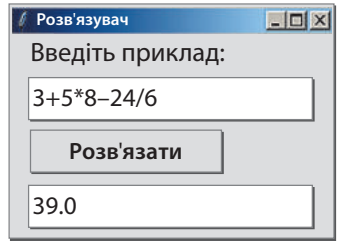


Рис. 19.8

2) Додайте віджет lab класу Label:

```
lab = Label(root, text = 'Введіть
приклад:', font = 'Arial 18')
lab.place(x = 20, y = 10)
```

Додайте до вікна два віджети класу Entry. Віджет entry1 призначений для введення прикладу:

```
s = ''
entry1 = Entry(root, text = s, width = 16, font = 'Arial 18')
entry1.place(x = 20, y = 50) # Розміщення текстового поля у вікні
```

Віджет entry2 призначений для виведення відповіді:

```
entry2 = Entry(root, text = s, width = 16, font = 'Arial 18')
entry2.place(x = 20, y = 140) # Розміщення текстового поля у вікні
```

3) Додайте до вікна віджет b1 класу Button:

```
b1 = Button(root, text = 'Розв'язати', command = b1_click)
b1.place(x = 40, y = 90)
```

4) Створіть обробник події Натискання на кнопку для об'єкта b1:

```
def b1_click():
    vidp = eval(entry1.get())
    entry2.delete(0, END)
    entry2.insert(0, vidp)
```

Для обробки математичного виразу й обчислення результату використовується функція eval(), яка обробляє рядок клавіатурних символів так само, як і оболонка Python IDLE.

5) Збережіть програму з іменем Вправа19.

6) Обчисліть значення виразів: а) $2 + 32$; б) $\frac{4}{5} - \frac{1}{3}$.

Комп'ютерне тестування

Виконайте тестове завдання 19 із автоматичною перевіркою результату.

Практична робота 6

Створення програм із графічним інтерфейсом

Завдання: скласти програму Калькулятор, що дає можливість виконувати будь-яку арифметичну дію над двома числами, що вводяться з клавіатури.

Обладнання: комп'ютер, середовище програмування IDLE.

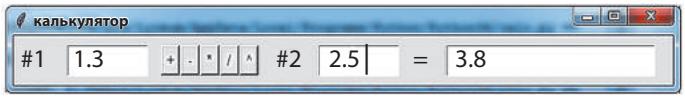
Хід роботи

Під час роботи з комп'ютером дотримуйтеся правил безпеки.

У Python IDLE виберіть команду File → New File.

1. Запишіть команду завантаження модуля tkinter і створіть вікно програми (650 × 50) із заголовком Калькулятор.

2. Додайте до вікна три віджети lab1–lab3 класу Label для виведення заголовків #1, #2, =. Задайте такі значення атрибутів об'єктів, щоб віджети виглядали так, як на рисунку.



3. Додайте до вікна віджет entry1 класу Entry для введення першого числа:

```
entry1 = Entry(root, text = 's1', width = 6, font = 'Arial 18')
entry1.place(x = 55, y = 10)
```

4. Додайте до вікна два віджети entry2 і entry3 класу Entry.

5. Додайте до вікна віджет b1 класу Button:

```
b1 = Button(root, text = '+', command = b1_click)
b1.place(x = 150, y = 10)
```

6. Додайте до вікна ще чотири об'єкти b2–b5 класу Button. Атрибуту text кожного з об'єктів задайте значення відповідно до зразка.

7. Створіть обробник події Натискання на кнопку b1_click.

У коді методу зчитуються значення з полів об'єктів entry1 та entry2, формується текстовий рядок s; функція eval() обробляє рядок символів і повертає значення виразу. Поле об'єкта entry3 очищується, і до нього виводиться результат.

```
def b1_click():
    s = entry1.get() + '+' + entry2.get()
    vidp = eval(s)
    entry3.delete(0, END)
    entry3.insert(0, vidp)
```

8. Створіть обробники події Натискання на кнопку b2_click–b5_click, вказуючи операцію, що відповідає заголовку кнопки.

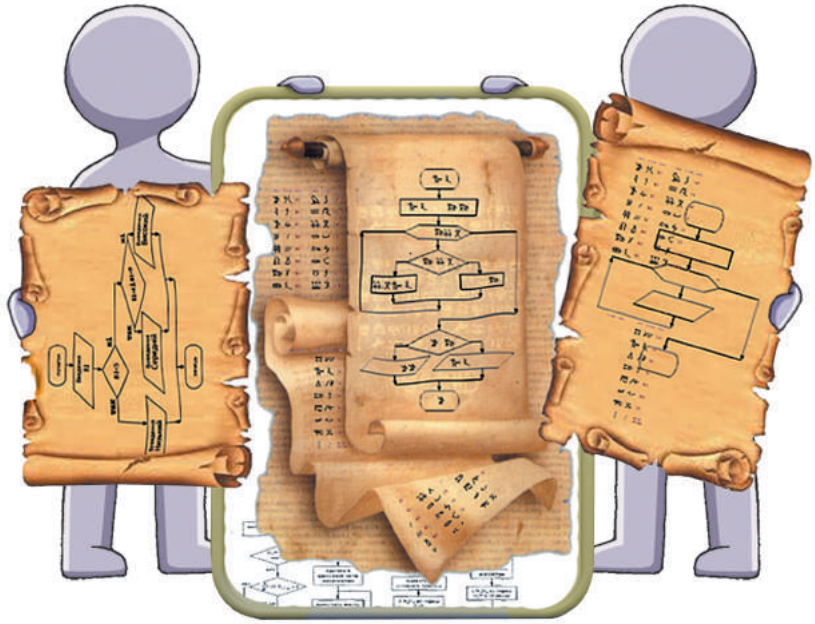
9. Збережіть програму у файлі з назвою Calc.

10. Виконайте програму для різних значень операндів.

Зробіть висновок: як створювати елементи графічного інтерфейсу та організувати діалог із програмою.

РОЗДІЛ 3
АЛГОРИТМИ ТА ПРОГРАМИ

3.2. АЛГОРИТМИ З ПОВТОРЕННЯМИ ТА РОЗГАЛУЖЕННЯМИ. ФУНКЦІЇ



ПОВТОРЮЄМО

Ви знаєте, що під час конструювання алгоритмів використовуються три базові алгоритмічні структури: *слідування*, *розгалуження*, *повторення*.

У 5 класі ви навчилися складати програми мовою Python з використанням *операторів розгалуження* та *операторів повторення*.

Ви дізналися, що оператор розгалуження можна використовувати в повній і неповній формах. А для опису дій, які мають виконуватися кілька разів, можна використовувати оператори циклу двох видів — і*з параметром* і *з передумовою*.

1. Що таке умова в операторах розгалуження та повторення?
2. Яке значення мають відступи команди від лівого краю вікна програми?
3. Поясніть структуру й правила виконання циклу з параметром; циклу з передумовою.
4. У чому полягає відмінність у використанні циклу з параметром і циклу з умовою?

Опанувавши матеріал, ви навчитеся організовувати множинне розгалуження. Ви розглянете правила вкладення циклів, навчитеся розв'язувати задачі способом перебору варіантів, складні задачі методом поділу на підзадачі та ін.

§ 20. Вкладені алгоритмічні структури розгалуження

Оператори розгалуження

Для реалізації розгалуження є умовні оператори if і if...else. Згадаємо синтаксис оператора розгалуження:

Неповна форма розгалуження	Повна форма розгалуження
if <умова> : —<оператор	if <умова> : —<оператор 1> else: —<оператор 2>
— обов'язковий відступ від лівого краю.	

Умова — це логічний вираз, значенням якого є True (Істина) або False (Хибність).

1 Перевіримо, чи є число x додатним.

Неповна форма розгалуження	Повна форма розгалуження

if x > 0:
  print ('Число додатне')
	if x > 0:
	  print ('Число додатне')
	else:
	  print ('Число не додатне')

Якщо умова хибна (x ⩽ 0), то керування передається оператору, наступному за if	Якщо умова хибна (x ⩽ 0), то виконується гілка Ні і виводиться повідомлення 'Число не додатне'

Проста умова утворюється за допомогою операцій відношення, *складена* умова — з кількох простих умов за допомогою логічних операцій AND (логічне І), OR (логічне АБО), NOT (логічне заперечення).

2 Визначимо значення складеної умови x >= 10 and x <= 18.

Значення x	5	20	10	15
Значення складеної умови	Хибне	Хибне	Істинне	Істинне

Вкладені розгалуження

Якщо після перевірки деякої умови виникає потреба знову робити вибір, застосовують *вкладені* розгалуження: в умовному операторі if по гілці Так або гілці Ні знову використовують оператор if. Розглянемо задачу.

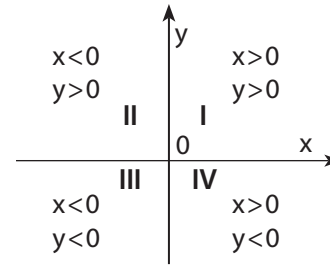


Рис. 20.1

Визначити N — номер координатної чверті, в якій міститься точка з координатами x, y ($x \neq 0$, $y \neq 0$) (рис. 20.1).

Складемо блок-схему алгоритму визначення N (рис. 20.2).

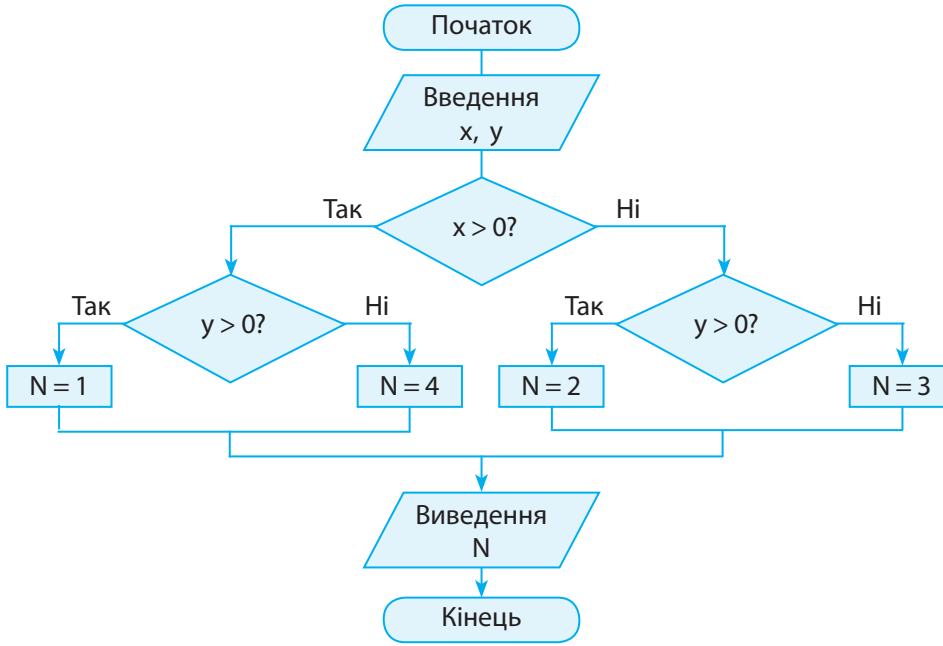


Рис. 20.2

Після визначення знака x знову з'являється потреба робити вибір залежно від знака y, тобто два наступних розгалуження «вкладаються» в перше.

За цим алгоритмом запишемо програмний код:

```
x = int(input('x = ?'))
y = int(input('y = ?'))
if x>0:
    if y>0: n = 1
    else: n = 4
else:
    if y>0: n = 2
    else: n = 3
print('N =', n)
```

Якщо $x = 5$, $y = -2$, то вираз $x > 0$ набуває значення True. По гілці Так перевіряється умова $y > 0$, яка при $y = -2$ набуває значення False, тому змінна n набуває значення 4.

Команди, вкладені в гілки оператора if, об'єднуються в блоки за величиною відступів. Відступ може бути будь-яким, головне, щоб у межах одного вкладеного блоку він був однаковий.

Множинне розгалуження

Якщо залежно від значення тієї чи іншої змінної може виконуватися одна з трьох (або більше) гілок програми, код стає громіздким і вкладати оператори if незручно.

Для таких випадків у Python є структура множинного розгалуження, яка реалізується оператором elif (скорочення від else if — «ще якщо»). У гілці elif обов'язково повинен бути логічний вираз, як у заголовку if. У кінці, після всіх elif, може використовуватися одна гілка else для обробки випадків, які не відповідають умовам гілки if і всіх elif.

3 Запрограмуємо поведінку гравця в лабіринті, якщо x — кількість монстрів, які зустрічаються на шляху.

```
x = int(input('x = ?'))
if x == 0:
    print('Монстрів немає. Шлях вільний!')
elif x<3:
    print('Стільки монстрів я легко подолаю')
```

```
elif x<5:
    print('Доведеться позмагатися')
else:
    print('Час рятуватися втечею')
```

Інструкція if-elif-else припиняє перегляд наступних гілок, як тільки логічний вираз у поточній гілці буде True. Так, якщо вираз при if (перша гілка) буде True, то після виведення рядка 'Монстрів немає. Шлях вільний!' виконання програми завершиться.

Таким чином, використання вкладених розгалужень допоможе запрограмувати вибір із великої кількості варіантів.

Питання для самоперевірки

1. Як записується і виконується умовний оператор у повній формі; неповній формі?
2. Якого значення набуде змінна a після виконання умовного оператора для початкових значень, наведених у таблиці?

Фрагмент програмного коду	Початкове значення a

if a%10 == 0:
a = a*2
elif a%10 == 3:
a = a+10
elif a%10 == 6:
a = a*10
else:
a = a+1
	13

	40
	88
	26

3. Поясніть схему виконання оператора if, у якому застосовані вкладені оператори розгалуження.
4. Напишіть програму, яка за кількістю правильних відповідей K визначає оцінку, яку учень отримав за виконання тесту, за правилом: якщо $K > 9$, то оцінка «Відмінно»; якщо $8 \leqslant K \leqslant 9$ — оцінка «Добре»; якщо $6 \leqslant K \leqslant 7$ — оцінка «Задовільно»; якщо $K < 6$ — повідомлення «Слід повторити роботу»).

Вправа 20

Касир продає квитки на автобус, який курсує від міста A до міста B. Вартість одного квитка залежить від відстані, на яку потрібно їхати пасажиру. Знайдіть вартість N квитків до населеного пункту, відстань до якого вводиться з клавіатури, якщо:

$$x = \begin{cases} 5 \text{ грн, до } 50 \text{ км;} \\ 15 \text{ грн, від } 51 \text{ до } 100 \text{ км;} \\ 25 \text{ грн, від } 101 \text{ до } 150 \text{ км;} \\ 35 \text{ грн, від } 151 \text{ км.} \end{cases}$$

У Python IDLE виберіть команду File → New File.

1) Запишіть оператор введення значення відстані й присвоювання цього значення змінній v:
```
v = int(input('v = ?')
```
2) Запишіть оператор введення значення змінної N:
```
N = int(input('N = ?')
```
3) Запишіть оператор if-elif-else, у якому залежно від значення v змінна x (вартість квитка) набуває значення згідно з умовою задачі:
```
if v<50: x = 5
elif v<100: x = 15
elif v<25: x = 25
else: x = 35
```
4) Знайдіть значення c — вартості N квитків. Виведіть значення c у консоль.
5) Збережіть файл з іменем Вправа20. Запустіть програму на виконання. Проаналізуйте результат виконання програми у вікні консолі.

6) Випробуйте програму для різних значень змінної v.

Комп'ютерне тестування

Виконайте тестове завдання 20 із автоматичною перевіркою результату.

§ 21. Вкладені алгоритмічні структури повторення

Ви ознайомилися з операторами циклу двох видів: із параметром та з умовою. Згадаємо правила запису команд повторення.

Цикл for

Якщо відома кількість повторень, зручно використовувати цикл for.

Синтаксис циклу for:

```
for x in range(start, stop, step):
    <тіло циклу>
```

— обов'язковий відступ від лівого краю

Змінна x є параметром (лічильником) циклу. Вбудована функція range визначає, скільки разів буде повторено виконання операторів тіла циклу. Ключове слово in наказує Python по черзі надати змінній x всі значення в діапазоні від start до stop–1 із кроком step. Оператори тіла циклу записують із відступом.

1. Надрукувати числа від 20 до 24. Якщо step = 1, цей параметр можна не вказувати.

```
for i in range(20, 25):
    print(i)
```

2. Надрукувати числа від 0 до 3. Якщо start = 0, цей параметр можна не вказувати.

```
for i in range(4):
    print(i)
```

3. Надрукувати парні числа в діапазоні від 10 до 20.

```
for i in range(10, 20, 2):
    print(i)
```

4. Надрукувати числа від 5 до 1. Якщо потрібно вести відлік у зворотному порядку, step має бути від'ємним.

```
for i in range(5, 0, -1):
    print(i)
```

5. Надрукувати числа зі списку значень [2, 9, 5, 8, 11]. Параметри можуть послідовно набувати значень зі списку.

```
b = [2, 9, 5, 8, 11]
for x in b:
    print('=>', x)
```

Цикл while

Цикл while (поки) буде повторюватися, поки виконується задана умова — її називають **умовою циклу.** Вона набуває значення True або False.

Синтаксис оператора:

```
while <умова>:
    <тіло циклу>
```

Тут <умова> — логічний вираз, що є умовою виконання циклу; якщо умова істинна, то виконуються оператори тіла циклу й керування повертається на перевірку умови. Якщо умова хибна, то виконується оператор, який є наступним після оператора while.

Блок-схему оператора while подано на рис. 21.1. Як бачимо, умова перевіряється щоразу на початку циклу.

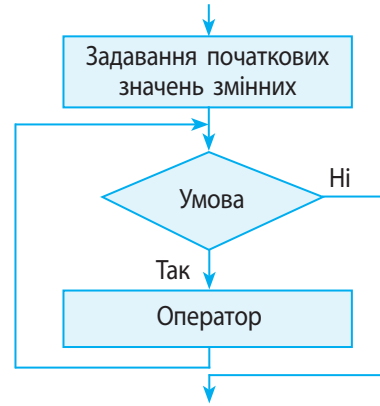


Рис. 21.1

6 Створити програму, яка пропонує ввести пароль. Якщо введено правильний пароль (наприклад, «секрет»), то цикл while припиняє роботу, якщо неправильний, то повторює запит.

```
password = ''
while password != 'секрет':
    print('Введіть пароль:')
    password = input()
print('Так, пароль ' + password)
```

Приклад діалогу з програмою наведено на рис. 21.2.

```
Введіть пароль:
аааа
Введіть пароль:
таємниця
Введіть пароль:
секрет
Так, пароль секрет
```

Рис. 21.2

Якщо введено правильний пароль, то виконується оператор print(), який знаходиться після циклу while. Якщо введено неправильний пароль, текст підтвердження не з'явиться, оскільки цикл продовжує роботу. Такий цикл може призвести до зациклювання. Щоб цього не сталося, треба виконавцю надати певну кількість спроб (наприклад, $\geqslant 3$) і припинити роботу циклу.

Вкладені цикли

Нехай нам потрібно засобами «черепашачої» графіки відтворити малюнок, наведений на рис. 21.3.

Рис. 21.3

Завантажимо команди для роботи з Черепашкою і встановимо ширину сліду Черепашки в 3 пікселі:

```
from turtle import*
width(3)
up()                                                    # Перо підняте
colors = ['yellow', 'blue', 'green', 'purple', 'red']   # Список кольорових
                                                          констант
```

Щоб намалювати один п'ятикутник, потрібно виконати цикл:

```
for n in range(5):
    color(colors[n])
    forward(20)
    left(360/5)
```

Щоб створити такий малюнок, потрібно малювання п'ятикутника повторити 10 разів (рис. 21.4).

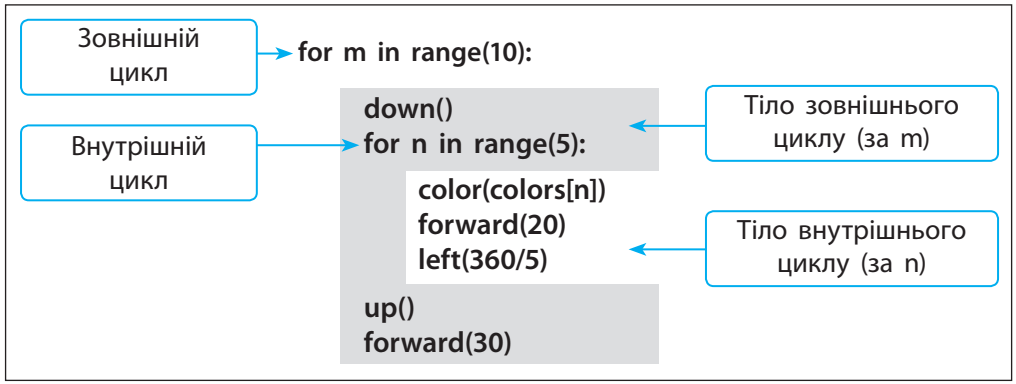


Рис. 21.4

Цикл називають **вкладеним**, якщо він міститься в тілі іншого циклу (його також називають **внутрішнім**), а цикл, у якому він міститься, — **зовнішнім**.

Вкладені цикли організовані таким чином: внутрішній цикл повністю вміщується в тілі зовнішнього циклу (рис. 21.5). Тут A — зовнішній цикл, B — внутрішній.

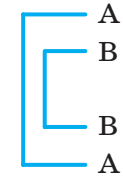


Рис. 21.5

Як внутрішній, так і зовнішній цикли можуть бути циклами з параметром або з умовою.

Порядок виконання вкладених циклів:

При першій ітерації (повторенні) зовнішнього циклу викликається внутрішній, який виконується до свого завершення.

Після цього керування передається в тіло зовнішнього циклу.

При другій ітерації зовнішнього циклу знову викликається внутрішній. І так триватиме доти, поки не завершиться зовнішній цикл.

7 Переконаємося, що зовнішній цикл перебирає список чисел (num_list), а внутрішній — список символів (alpha_list).

```
num_list = [1, 2, 3]
alpha_list = ['a', 'b', 'c']
for number in num_list:
    print(number)
    for letter in alpha_list:
        print(letter)
```

При першій ітерації зовнішнього циклу в консоль виводиться 1, а потім викликається внутрішній цикл, у якому виводяться послідовно a, b, c. Після цього керування передається на початок зовнішнього циклу, виводиться 2, а потім знову виконується внутрішній цикл і виводяться a, b, c і т. д.

! Параметрами вкладених циклів for мають бути різні змінні.

Як бачимо, вкладені цикли можуть бути дуже корисними для організації перебору чисел у певному діапазоні під час створення малюнків, зокрема орнаментів, тощо.

Питання для самоперевірки

1. Які види циклів ви знаєте? У яких випадках застосовують кожний із цих циклів?
2. У чому полягає правило вкладення циклів?
3. Проаналізуйте циклічну конструкцію:

```
for i in range(1, 2):
    for j in range(1, 3):
        for k in range(1, 3):
            print(i, j, k)
```

а) Назвіть тіло кожного циклу.
б) Скільки разів буде виконаний кожний цикл?
в) Який результат буде виведений після виконання програми?

4. Що буде надруковано в результаті виконання фрагмента програми?

а)
```
for i in range(1, 3):
    for j in range(2, 10, 2):
        print(i, j)
```

б)
```
for i in range(1, 4):
    for j in range(1, 4):
        print (i, '*', j, '=', i*j)
```

5. Складіть програму для обчислення значення виразу $2k + n$ при всіх заданих значеннях змінних: $n = 1$, $n = 2$, $n = 3$ і $k = 2$, $k = 4$, $k = 6$, $k = 8$.
6. У 2019 році обсяг деревини на території лісового масиву становив 120 000 м^3. Щоб заготовити деревину, щорічно вирубується 9500 м^3 лісу. Щорічний природний приріст становить 5,5 %. Напишіть програму для визначення обсягу деревини, що залишиться в лісовому масиві через n років.

Вправа 21

Написати програму малювання орнаменту (рис. 21.6).

Рис. 21.6

У Python IDLE виберіть команду File → New File.

1) Завантажте бібліотеки модуля turtle і визначте початкові параметри малювання:

```
from turtle import*
width(3)
up()
x = 0
y = 0
colors = ['red', 'yellow', 'blue', 'green', 'orange']
```

2) Для малювання одного фрагмента потрібно намалювати 5 вкладених кіл:

```
for j in range(5):
    down()
    color(colors[j])
    circle(25–j*5)    # Радіус зменшується на j*5 пікселів
    up()
    left(90)
    forward(5)
    right(90)
```

3) Для малювання орнамента потрібно повторити малювання фрагмента 8 разів:

```
for i in range(8):
    up()
    goto(i*50, 0)     # Зсув пера праворуч на 50 пікселів
# Внутрішній цикл малювання 1 фрагмента
```

4) Збережіть файл з іменем Вправа21. Виконайте програму.
5) Додайте до списку кольорів колір 'purple'. Змініть програму так, щоб вона малювала в кожному фрагменті 6 кіл.
6) Змініть програму так, щоб малювання фрагмента повторювалось 10 разів.

Комп'ютерне тестування

Виконайте тестове завдання 21 із автоматичною перевіркою результату.

§ 22. Розв'язування задач за допомогою вкладених циклів

На уроках математики корінь рівняння $3x + 4 = 15$ знаходять у такий спосіб: $x = \frac{16 - 4}{3}$. Існує й інший спосіб пошуку невідомої величини — **метод перебору значень**, який часто використовується в програмуванні. Наприклад, його використовують, якщо потрібно знайти цілий корінь деякого складного рівняння.

При цьому перебираються можливі цілі значення x із обмеженого діапазону і підставляються в рівняння. Відповіддю є таке значення x, при якому рівняння перетворюється на рівність.

Розглянемо, як застосовується метод перебору значень.

1 Знайдемо сторони прямокутника, периметр якого 20 см, одна зі сторін більша за іншу на 4 см. Позначимо сторони змінними a і b. Тоді $b = a + 4$, а периметр: $2 \cdot (a + b) = 2 \cdot (2a + 4)$.

Зрозуміло, що значення a може бути більшим за 0, але меншим від 10 (половини периметра). У циклі переберемо значення a зі вказаного діапазону.

Отримаємо значення a, при якому $2 \cdot (2a + 4) = 20$:

```
for a in range(10):
    if 2*(2*a+4) == 20:
        print('a = ', a, 'b = ', a+4)
```

Якщо потрібно знайти дві або більше невідомі величини методом перебору, використовується вкладення циклів.

2 Складемо програму для розв'язування задачі. Визначимо, скільки фазанів і кролів, якщо всього в клітці 35 голів і 94 ноги.

Нехай x — кількість фазанів, y — кількість кролів. Тоді $x + y = 35$. Відомо, що у фазанів 2 ноги, у кролів — 4, тому $2x + 4y = 94$. Ці дві умови мають виконуватись одночасно, тобто утворюють складену умову. Фазанів може бути від 0 до 35, кролів теж може бути від 0 до 35.

Організуємо перебір варіантів:

```
for x in range(36):
    for y in range(36):
        if ((2*x+ 4*y == 94)and (x+y==35)): print('x=',x, ' y=',y)
```

Отримаємо єдину відповідь: $x = 23$; $y = 12$.

3 Знайдемо всі трицифрові натуральні числа, сума цифр яких дорівнює їх добутку, та визначимо кількість таких чисел. Можна перебрати всі можливі сполучення цифр, із яких утворюється десятковий запис трицифрового числа, і перевірити умову задачі для кожного сполучення.

```
k = 0
for a in range (1, 10):      # Першими можуть бути цифри від 1 до 9
    for b in range (10):           # На другому й третьому місцях
        for c in range (10):         можуть стояти цифри від 0 до 9
            if a+b+c == a*b*c:
                k = k+1
                print(100*a+10*b+c)
print('k = ', k);
```

У коді цикл із параметром c є тілом циклу з параметром b. А він, у свою чергу, є тілом циклу з параметром a.

Питання для самоперевірки

1. Що буде надруковано в результаті виконання фрагмента програми?

а)
```
for i in range(1, 10):
    for j in range(1, 10):
        print(i, '*', j, '=', i*j)
```

б)
```
for i in range(20):
    j = 2
    print('i =', i)
    while(j <= (i/2)):
        if i%j == 0: print(j)
        j = j+1
```

в)
```
alpha_list = ['a', 'b', 'c']
for letter1 in alpha_list:
    for letter2 in alpha_list:
        print(letter1, letter2)
```

г)
```
for i in range(20):
    j = 2
    prap = 0
    while (j <= (i/2)):
        if i%j == 0: prap = 1
        j = j+1
    if prap == 0: print(i)
```

2. Складіть програму для розв'язування задачі.

1) Знайдіть усі двоцифрові числа, які діляться на добуток їхніх цифр.

2) Скільки в зоопарку страусів та верблюдів, якщо разом у них 40 ніг?

Вправа 22

Написати програму для пошуку розв'язку задачі.

Плата за одного бика становить 20 карбованців (крб), за корову — 10 крб, за теля — 1 крб. Скільки можна купити биків, корів і телят, якщо на 200 крб треба купити 100 голів худоби?

1) Позначимо літерою b кількість биків; k — кількість корів; t — кількість телят. Загальна кількість голів дорівнює 100, тоді $b + k + t = 100$.

За биків заплатили $20b$ крб, за корів — $10k$ крб, за телят — t крб, отже, $20b + 10k + t = 200$.

На 200 крб можна купити:

- не більше ніж 10 биків, тобто $0 <= b <= 10$;
- не більше ніж 20 корів, тобто $0 <= k <= 20$;
- не більше ніж 200 телят, тобто $0 <= <= t <= 200$.

Таким чином, необхідно перебрати всі можливі значення змінних b, k, t і вивести в консоль той набір значень, для яких виконується умова (20*b+10*k+t = 200) and (b+k+t = 100).

2) Напишіть оператори циклу для перебору всіх можливих сполучень значень b, k, t. Для кожного сполучення значень потрібно перевіряти умову задачі.

```
for b in range (11):
    for k in range (21):
        for t in range (201):
            if (20*b+10*k+t == 200) and (b+k+t == 100):
                print('Биків', b)
                print ('корів', k)
                print ('телят', t)
```

3) Збережіть файл з іменем Вправа22.

4) Запустіть програму на виконання, проаналізуйте результат виконання програми у вікні консолі.

5) Внесіть зміни до програмного коду для розв'язування задачі: в магазині придбали олівці по 6 грн і зошити по 5 грн. Вартість всієї покупки склала 68 грн. Скільки купили олівців і зошитів?

6) Збережіть файл і запустіть програму на виконання. Проаналізуйте результат виконання програми у вікні консолі.

Комп'ютерне тестування

Виконайте тестове завдання 22 із автоматичною перевіркою результату.

Практична робота 7

Створення програм із розгалуженнями і повтореннями

Завдання: скласти програму для розв'язування рівнянь із двома невідомими методом перебору.

Обладнання: комп'ютер зі встановленим середовищем програмування Python.

Хід роботи

Під час роботи з комп'ютером дотримуйтеся правил безпеки.

У Python IDLE виберіть команду File → New File.

1. Програма має знаходити такі значення x, y, за яких виконується умова $ax + by = c$. Запишіть оператор введення значення змінної a:

```
a = int(input('a = ?'))
```

2. Запишіть оператори введення значень змінних b, c.

3. Змінні x, y можуть набувати значень із обмеженого діапазону. За умовою задачі потрібно визначити, яких най-

більших значень $d1$ і $d2$ можуть набувати ці змінні. Запишіть оператор введення значення змінної $d1$:
d1 = int(input('Яке найбільше значення може набувати X?'))

4. Запишіть оператор введення значення змінної $d2$.

5. Рівняння може мати більше від однієї пари розв'язків, тому програма має повідомляти, скільки пар значень x, y, що задовольняють умову, знайдено. Створіть змінну k для збереження значення кількості знайдених пар значень x, y; перед початком перебору змінній k надається значення 0: k = 0

6. Запишіть оператори циклу для перебору всіх можливих значень змінних x, y. Для кожного сполучення значень потрібно перевіряти виконання умови $ax + by = c$. Значення x потрібно перебирати в діапазоні range(d1), значення y — у діапазоні range(d2). Якщо в тілі внутрішнього циклу виконується умова $ax + by = c$, то лічильник знайдених відповідей збільшується на 1, і поточні значення параметрів циклів x і y виводяться до консолі:

```
for x in range(d1):
    for y in range(d2):
        if a*x+b*y == c:
            k = k+1
        print ('x = ',x, ' y = ',y)
```

7. Після закінчення роботи циклів потрібно вивести кількість знайдених розв'язків:
print ('Знайдено відповідей', k)

8. Збережіть програмний код з іменем Практична робота7.

9. Виконайте програму для рівняння $22x + 13y = 1000$.

10. За допомогою створеної програми розв'яжіть задачу.
Діти зібрали 174 кг макулатури. Кожен хлопчик зібрав по 21 кг, а кожна дівчинка — по 15 кг. Скільки хлопчиків і дівчаток збирали макулатуру?

Зробіть висновок: як створювати програми з повтореннями та розгалуженнями для розв'язування задач методом перебору.

§ 23. Розв'язування задач методом поділу на підзадачі

Метод поділу складної задачі на підзадачі

Припустимо, нам потрібно створити малюнок. Спочатку ми маємо визначити, що саме потрібно намалювати (будинок, дерево, сонце тощо), та скласти алгоритм малювання (рис. 23.1).

Визначивши основні кроки алгоритму малювання, почнемо планувати дії щодо їх реалізації: як намалювати стіни, дах та ін.

Метод поділу на підзадачі полягає в тому, щоб у ході розроблення алгоритму розв'язування складної задачі спочатку визначити основні кроки, тобто розробити стратегію розв'язування задачі (*що зробити?*).

Далі потрібно розробити шляхи та методи реалізації намічених кроків (*як зробити?*).

Алгоритм розв'язування кожної підзадачі можна оформити як **функцію** — окремий іменований блок програмного коду (рис. 23.2).

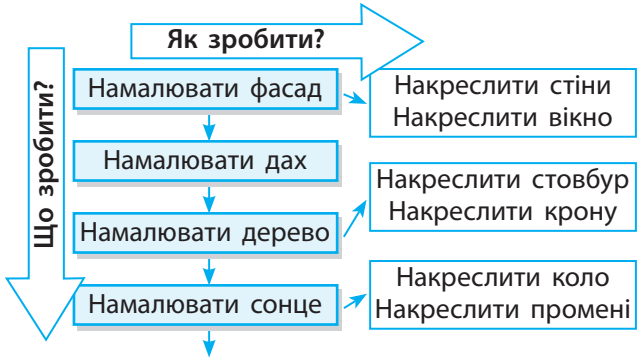


Рис. 23.1

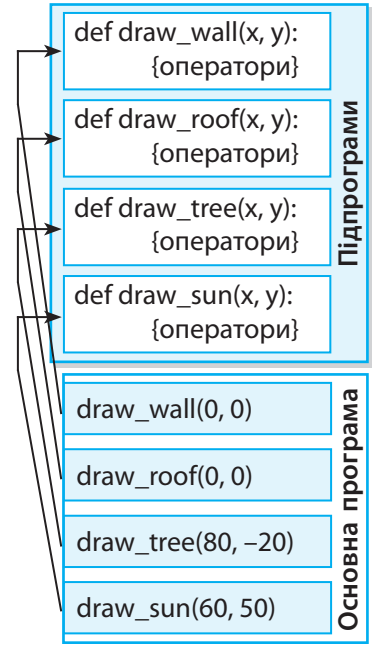


Рис. 23.2

Функції реалізують алгоритми розв'язування підзадач складної задачі. Їх можна порівняти з невеликими програмами, що вбудовані в основну програму.

Для малювання зображення, наведеного на рис. 23.3, перед основним програмним кодом створюються чотири функції з іменами draw_wall, draw_roof, draw_tree, draw_sun для розв'язування підзадач основної задачі.

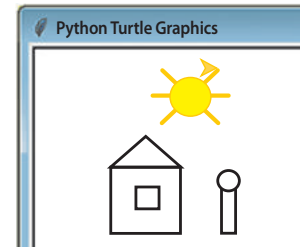


Рис. 23.3

В основній програмі ці функції викликаються послідовно за їхніми іменами.

Функції роблять програмний код більш коротким, зручним для читання і, головне, — придатним для повторного використання. Наприклад, якщо ми захочемо намалювати ще одне дерево біля будинку, то повторно викличемо функцію draw_tree з іншими координатами початку малювання.

Правила створення і виклику функцій у Python

Ми вже ознайомились зі стандартними функціями різного призначення. Наприклад, функція randint() повертає випадкове число, функція print() виводить список значень у вікно консолі, функція forward() пересуває Черепашку. Ці функції вбудовані, тобто описані в модулях і бібліотеках мови Python.

У Python є можливість створювати й використовувати власні функції, так звані **функції користувача** (під користувачем розуміють програміста — того, хто пише програму, а не того, хто потім її використовує). Розглянуті раніше функції draw_roof, draw_tree, draw_sun є функціями користувача (текст програми малювання рис. 23.3 наведено на сайті interactive.ranok.com.ua).

Функція складається з трьох частин: *імені*, *параметрів* і *тіла*.

У Python функції визначаються за допомогою зарезервованого слова def.

Опис функції:

```
def ім'я_функції(<перелік параметрів>):
    тіло функції
```

Тіло функції являє собою послідовність операторів, які будуть виконані після викликання функції.

Ознайомимося з правилами створення функцій у Python.

- Блок функції починається з ключового слова def, після якого пишуть назву функції й круглі дужки ().
- Усередині дужок пишуть *параметри* — імена змінних, які отримують значення під час виклику функції. Ці параметри ще називають формальними. Якщо функції не треба передавати значення, необхідно записати порожні дужки.
- Після дужок ставиться двокрапка і з нового рядка з відступом записуються оператори тіла функції.

Після того як функцію створено, її можна викликати з іншої функції або безпосередньо з оболонки Python.

Щоб **викликати функцію**, потрібно:

1) ввести ім'я функції й додати дужки;
2) у дужках перелічити *аргументи функції* — значення, які треба присвоїти формальним параметрам.

Якщо формальних параметрів немає, то при виклику після імені функції потрібно писати порожні дужки.

Під час виклику відбувається виконання команд тіла функції.

> ! Опис функції має міститися вище від виклику функції. Це пояснюється тим, що інтерпретатор зчитує код порядково, і про те, що міститься в рядках, які розташовані далі, йому невідомо.

Якщо виклик функції записано раніше, ніж її опис, то виникає помилка NameError.

1 Створимо і викличемо функцію, яка виводить у консоль квадрат числа.

Призначення операторів у цьому програмному коді пояснюється на рис. 23.4.

```
def kv(x):          → заголовок функції
    print (x*x)     → тіло функції
kv(5)               → виклик функції
```

Рис. 23.4

2 Опишемо і викличемо функції для малювання будинку (див. рис. 23.3). Під час виклику функцій у дужках указуються значення координат x, y, із яких буде починатися малювання:

```
def draw_wall(x, y): # Фасад
    up()
    goto(x, y)
    down()
    for n in range(4):    # Стіни
        forward(50)
        right(90)
    up()
    goto(x+15, y–15)
    down()
    for n in range(4):    # Вікно
        forward(20)
        right(90)
def draw_roof(x, y): # Дах
    up()
    goto(x, y)
    down()
    left(50)
    forward(40)
    right(100)
    forward(40)
    up()
    left(50)
# Основна програма
width(3)
draw_wall(0, 0)
draw_roof(0, 0)
```

Таким чином, за допомогою функцій можна поділити програму на окремі частини, кожна з яких виконує конкретне завдання. Функції також надають можливість багаторазово викликати один і той самий код із різних місць програми.

Питання для самоперевірки

1. На які підзадачі можна поділити задачу малювання жабки (рис. 23.5)?
2. Чи можна описати функцію, жодного разу її не викликавши?
3. Напишіть функцію для малювання квадрата. Як можна використати цю функцію для малювання квадратів, наведених на рис. 23.6? Чи простіше експериментувати з програмою, в якій виконання підзадач оформлене як функції?

Рис. 23.5

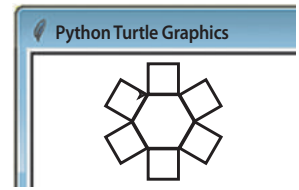


Рис. 23.6

4. У наведеному програмному коді назвіть:
а) заголовок функції;
б) тіло функції;
в) оператор виклику функції.

```
def sum(a, b):
    print (a+b)
sum(3, 5)
```

Вправа 23

Написати програму малювання вулиці з кількома будинками з огорожею (рис. 23.7).

Рис. 23.7

1) У написанні коду Малювання вулиці використайте функції draw_wall, draw_roof для малювання будинку.
2) Опишіть функцію draw_fence(x, y) для малювання фрагмента огорожі.
3) Напишіть програму малювання вулиці з трьох будинків:

```
from turtle import*
# Додайте сюди описи функцій
width(3)
x = 0
for n in range(3):
    draw_wall(x, 0)
    draw_roof(x, 0)
    x = x+60     # Відстань між початком малювання будинків
```

4) Додайте до програми малювання 20 фрагментів (рис. 23.8).

```
x = –10
y = –50
for n in range(20):
    draw_fence (x, y)
    x = x+10          # Відстань між фрагментами огорожі
```

Рис. 23.8

5) Збережіть файл з назвою Вправа23 і виконайте програму.
6) Поекспериментуйте з кількістю будинків і довжиною огорожі.

Комп'ютерне тестування

Виконайте тестове завдання 23 із автоматичною перевіркою результату.

§ 24. Обчислення з використанням функцій користувача

Повернення результату виконання функції

Ви вже вмієте викликати функції, передаючи значення для їхніх формальних параметрів. Щоб викликати функцію, що не повертає значення, слід ввести її ім'я й додати дужки зі списком значень, які треба присвоїти формальним параметрам.

Припустимо, що в тілі функції виконуються деякі обчислення. Якщо результат обчислень у подальшому треба використовувати в основній програмі, у тілі функції потрібно записати оператор: return <вираз>.

```
def avr(c1,c2):
    return (c1+c2)/2
sar1 = avr(x1,x2)
```

Рис. 24.1

Оператор return припиняє виконання функції й повертає значення виразу на місце виклику функції (рис. 24.1).

Ім'я і параметри функції слід записати у правій частині оператора присвоєння. Можна також викликати функцію зі списку виведення оператора print().

1 Складемо програму для розв'язування задачі.

У першому магазині мобільні телефони двох видів коштують $x1$, $x2$ грн, а в другому — $y1$, $y2$ грн. Визначимо, у якому магазині середня ціна телефонів нижча.

```
def avr(c1, c2):
    sar = (c1+c2)/2
    return sar
x1 = int(input('x1 = ?'))
x2 = int(input('x2 = ?'))
sar1 = avr(x1, x2)
y1 = int(input('y1 = ?'))
y2 = int(input('y2 = ?'))
sar2 = avr(y1, y2)
```

```
if sar1<sar2: print('Нижчі ціни у I магазині')
elif sar1>sar2: print('Нижчі ціни у II магазині')
else: print('Ціни однакові')
```

Проаналізуємо програмний код.

- В операторі sar1 = avr(x1, x2) функція avr викликається з фактичними параметрами x1, x2. Формальному параметру c1 присвоюється значення x1, параметру c2 — значення x2. Значення, що повертається, присвоюється змінній sar1.
- В операторі sar2 = avr(y1, y2) функція avr викликається з параметрами y1, y2. Значення, що повертається, присвоюється змінній sar2.

Області видимості змінних

Не можна використати змінні, створені в тілі функції, після того, як ця функція завершить роботу. Ці змінні існують тільки під час її виконання. У таких випадках говорять, що область видимості змінних обмежена.

Область видимості — це та частина програмного коду, в якій змінна доступна для використання.

Змінні, створені всередині тіла функції, є *локальними* — їх «не видно» з інших функцій і з основної програми.

Якщо змінна створена в основній програмі, вона є *глобальною* і її можна використовувати у всіх наступних командах і в будь-якій функції. Якщо потрібно змінити глобальну змінну всередині функції, то слід використовувати ключове слово global (рис. 24.2).

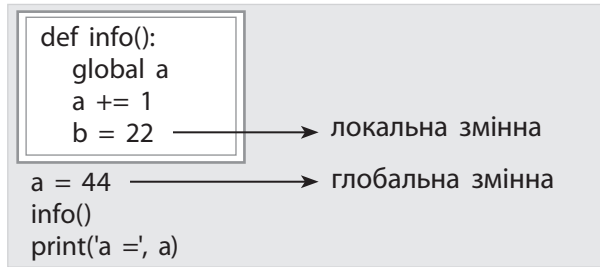


Рис. 24.2

Алгоритм створення функції в Python

Алгоритм знаходження найбільшого спільного дільника (НСД) двох чисел m і n НСД(m, n) описано в III ст. до н. е. в трактаті «Початки» грецького математика Евкліда: поки $a \neq b$, від більшого числа віднімати менше.

Розглянемо алгоритм створення функції для обчислення деякого значення на прикладі пошуку НСД двох чисел (рис. 24.3).

```
def nsd(m, n):
# Обчислення НСД 2 аргументів
    while m != n:
        if m>n:  m = m–n
        else:  n = n–m
        print(m,' ',n)
    return m
print ('nsd =', nsd(90, 54))
```

1. Визначити, що має робити функція, і дібрати належну назву
2. Написати тіло функції
3. Викликати функцію із зазначеними аргументами для перевірки роботи

Рис. 24.3

2 Виконаємо програму для $n = 90$; $n = 54$.

Результат її виконання наведено на рис. 24.4.

```
36   54
36   18
18   18
nsd = 18
```

Рис. 24.4

Варто супроводжувати функції коментарями: це полегшить подальшу роботу з програмою.

Питання для самоперевірки

1. У наведеному програмному коді назвіть заголовок функції, тіло функції, оператор виклику функції.

```
def sum(a, b):
    return a+b
s = sum(3, 5)
```

2. Що надрукує програма, якщо $x = 123$; $x = 54\,321$?

```
def f(x):
    k = 0
    while x>0:
        x = x // 10
        k += 1
```

```
        return k
x = int(input('x = ?'))
print(f(x))
```

3. Описано функцію для визначення більшого з двох чисел. Як використати цю функцію для визначення найбільшого з трьох чисел x, y, z?

```
def m(a, b):
    if a>b: k = a
    else: k = b
    return k
```

4. Створіть функцію, яка друкує добуток двох чисел.

Вправа 24

Дано функцію для обчислення найбільшого спільного дільника nsd(m, n). Користуючись нею, знайти найменше спільне кратне (nsk) чисел a, b, c, d, якщо відомо, що nsk(a, b) = a*b//nsd(a, b).

1) У підручнику описано функцію nsd(m, n) обчислення НСД чисел m, n. Запишіть у вікні програми код функції.
2) Запишіть у коді основної програми оператори для введення значень змінних a, b.
   Змінній nab надайте значення НСК чисел a, b:
   nab = a*b//nsd(a, b)
3) Запишіть у коді програми оператори для введення значень змінних c, d. Змінній nabc надайте значення НСК чисел nab, c: nabc = nab*c//nsd(nab, c)
4) Змінній nabcd надайте значення НСК чисел nabc, d.
5) Виведіть отримане значення nabcd.
6) Збережіть файл з іменем Вправа24. Виконайте програму:
   а) для набору чисел 25, 5, 10, 2;
   б) для власного тестового набору чисел.

Комп'ютерне тестування

Виконайте тестове завдання 24 із автоматичною перевіркою результату.

Практична робота 8
Створення програм із використанням функцій користувача

Завдання: скласти ігрову програму Спіймай кнопку.
Обладнання: комп'ютер зі встановленим середовищем програмування Python.

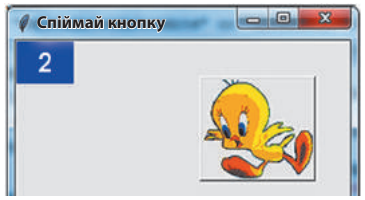
Хід роботи

Під час роботи з комп'ютером дотримуйтеся правил безпеки.

У Python IDLE виберіть команду File → New File.

1. Запишіть команди завантаження модулів:
```
from tkinter import*
from random import randint
import time      # Модуль для роботи з часом і таймерами
```

2. Створіть вікно програми розмірами 300 × 300 із заголовком Спіймай кнопку.



3. Додайте до вікна віджет lab класу Label. Задайте такі значення атрибутів об'єкта lab, щоб віджет виглядав так, як на рисунку.

4. Помістіть у вашу папку файл із розширенням .gif, що містить малюнок, який ви збираєтесь помістити на кнопку. У наведеному коді файл має ім'я 1.gif.

5. Додайте до вікна віджет b1 класу Button:
```
my_image = PhotoImage(file = '1.gif')
b1 = Button(root, command = b1_click, image = my_image)
b1.place(x = 40, y = 40)
```

6. Кількість натискань на кнопку зберігається у глобальній змінній clicks. Задайте її початкове значення: clicks = 0.

7. Створіть обробник події Натискання на кнопку b1_click. У цій функції змінюється глобальна змінна clicks, яка

зберігає кількість натискань, значення змінної clicks виводиться в заголовок напису lab. Під час кожного клацання кнопки спрацьовуватиме функція b1_click(), і кількість натискань відображатиметься в заголовку напису.

```
def b1_click():
    global clicks
    clicks += 1
    lab.config(text = str(clicks))
```

8. Для того щоб кнопка «стрибала» у вікні, змінюючи координати розташування випадковим чином через певні проміжки часу, створіть функцію clock():

```
def clock():
    x1 = randint(1, 300)
    y1 = randint(1, 300)
    b1.place(x = x1, y = y1)
    root.after(1000, clock)
```

Метод after виконує роль таймера: можна відкласти виконання деякого коду на певний час. Метод after приймає два аргументи: час у мілісекундах і функцію, яку слід виконати через певний час.

У наведеному коді метод after викликається для об'єкта root і має такі аргументи:

- час — 1000 мілісекунд (1 секунда);
- ім'я функції — clock, тобто через 1 секунду функція викличе сама себе.

9. В основній програмі запишіть команду виклику функції clock: clock().

10. Останнім рядком програмного коду запишіть оператор: root.mainloop()

11. Збережіть програмний код з іменем Gra, виконайте програму. Чи вдається вам спіймати кнопку? Якщо ні, збільште значення першого аргументу методу after.

Зробіть висновок: як створювати функції для розв'язування підзадач цієї задачі.

КОМП'ЮТЕРНИЙ СЛОВНИК

Алгоритм — скінченна послідовність команд для виконавця, яка чітко визначає, які дії та в якому порядку потрібно виконати для розв'язування певного завдання.

Алгоритм із повторенням — алгоритм, у якому передбачено повторення деяких команд.

Алгоритм із розгалуженням — алгоритм, у якому ті чи інші команди виконуються залежно від заданої умови.

Анімація — оснащення об'єкта ефектами руху та/або озвучування.

Блок-схема — графічне подання алгоритму у вигляді організованої послідовності блоків.

Величина — інформаційний об'єкт (число, символ, рядок тощо), основними характеристиками якого є назва, вид, тип і значення.

Відмова — подія, що виникає в разі виклику команди в неприпустимому для цієї команди стані середовища.

Гіперпосилання (або **посилання**) — це виділений кольором чи іншим чином текст, зображення чи кнопка, натиснення на які викликає перехід на перегляд інших даних.

Градієнт — плавний перехід від одного кольору до іншого.

Змінна — величина, значення якої може змінюватися в ході виконання програми.

Клон — копія об'єкта, яка зберігає зв'язок з оригіналом.

Комп'ютерна графіка — розділ інформатики, у якому вивчаються методи створення й опрацювання зображень за допомогою комп'ютера.

Комп'ютерна презентація — набір матеріалів, підготовлених для перегляду на екрані комп'ютера або на проекційному чи іншому екрані з метою подання презентаційного об'єкта.

Контейнер слайда — об'єкт, призначений для розміщення об'єктів слайда для їхнього можливого автоматичного форматування під час зміни макета та/або оформлення слайда.

Макет слайда — спосіб розташування об'єктів на слайді.

Мова програмування — штучна мова, що являє собою систему позначень і правил для запису алгоритмів у формі, придатній для подальшого їх виконання за допомогою комп'ютера.

Переходи слайда — засоби для встановлення й налаштування ефектної зміни слайдів.

Презентація — це сукупність дій (виступ, лекція тощо) і документів (фотографій, схем, аудіо- та відеозаписів, текстів тощо), призначених для донесення інформації про певну ідею, проект, результат роботи, товар, винахід тощо.

Програма — впорядкована послідовність команд для комп'ютера, виконання якої реалізує алгоритм розв'язування певної задачі.

Роздільна здатність (або **роздільність) зображення** — кількість крапок, що припадають на одиницю довжини растрового зображення.

Сайт (вебсайт) — кілька пов'язаних гіперпосиланнями вебсторінок, які мають спільну тему та розміщення.

Синтаксис мови — сукупність правил побудови команд мови програмування.

Система команд виконавця — повний перелік команд, які «розуміє» і може виконати виконавець.

Система опрацювання комп'ютерних презентацій — прикладне програмне забезпечення для створення й редагування презентацій на комп'ютері.

Слайд — екранна сторінка комп'ютерної слайдової презентації.

Тема слайда — іменована сукупність оформлення об'єктів на слайді: тла, шрифтів, графічних об'єктів тощо.

Тіло циклу — серія команд, які повторюються під час виконання циклу.

Алфавітний покажчик

Зміст

Відомості про користування підручником

№ з/п	Прізвище та ім'я учня / учениці	Навчальний рік	Стан підручника	
			на початку року	у кінці року
1				
2				
3				
4				
5				

Навчальне видання

БОНДАРЕНКО Олена Олександрівна
ЛАСТОВЕЦЬКИЙ Василь Васильович
ПИЛИПЧУК Олександр Павлович
ШЕСТОПАЛОВ Євген Анатолійович

«ІНФОРМАТИКА»
підручник для 6 класу закладів загальної середньої освіти

Рекомендовано Міністерством освіти і науки України

Видано за рахунок державних коштів. Продаж заборонено

Редактор *Л. А. Каюда.* Технічний редактор *А. В. Пліско.*
Художнє оформлення *В. І. Труфена.*
Комп'ютерна верстка *С. В. Яшиша.* Коректор *Н. В. Красна.*

Окремі зображення, що використані в оформленні підручника,
розміщені в мережі Інтернет для вільного використання

Підписано до друку 31.05.2019. Формат 70×90/16.
Папір офсетний. Гарнітура Шкільна. Друк офсетний.
Ум. друк. арк. 11,70. Обл.-вид. арк. 15,6. Тираж 65 596 прим. Зам. 3606-2019.

ТОВ Видавництво «Ранок»,
вул. Кібальчича, 27, к. 135, Харків, 61071.
Свідоцтво суб'єкта видавничої справи ДК № 5215 від 22.09.2016.
Адреса редакції: вул. Космічна, 21а, Харків, 61145.
E-mail: office@ranok.com.ua. Тел. (057) 719-48-65, тел./факс (057) 719-58-67.

Підручник надруковано на папері українського виробництва

Надруковано у друкарні ТОВ «ТРІАДА-ПАК»,
пров. Сімферопольський, 6, Харків, 61052.
Свідоцтво суб'єкта видавничої справи ДК № 5340 від 15.05.2017.
Тел. +38 (057) 712-20-00. E-mail: sale@triada.kharkov.ua

Властивості об'єктів

Об'єкт: ВОВК (природний)
- Колір: сірий
- Маса тіла: до 100 кг
- Довжина тіла: до 160 см
- Кількість лап: 4
- Довжина хвоста: до 52 см

Об'єкт: ВОВК (штучний)
- Матеріал: хутро
- Колір: сірий
- Маса: 0,5 кг
- Довжина: 30 см
- Кількість лап: 4

Об'єкт: ВОВК (віртуальний)
- Колір: сірий
- Положення: вертикальне
- Наявність хвоста: істина
- Кількість лап: 4
- Зріст: 110 см

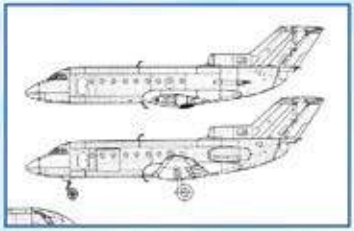
ПРОГРАМИ

Моделі реального об'єкта

Об'єкт: ЛІТАК

Модель: ІГРАШКА

- Призначення: перше знайомство



Модель: КРЕСЛЕННЯ

- Призначення: конструювання літака

Модель: МАКЕТ

- Призначення: перевірка властивостей

Модель: ТРЕНАЖЕР

- Призначення: підготовка пілотів

ІНФОРМАТИКА
6 клас

Особливості підручника:

- Мотивація навчальної діяльності
- Приклади практичного застосування набутих знань
- Система вправ для формування практичних навичок роботи за комп'ютером
- Питання для самоперевірки за кожною темою
- Покрокові описи практичних робіт

Інтернет-підтримка дозволить:

- здійснити онлайн-тестування за кожною темою
- ознайомитися з додатковими матеріалами до уроків

ВИДАВНИЦТВО
РАНОК

ISBN 978-617-09-5188-5
2315 0102
9 786170 951885

Інтернет-підтримка
interactive.ranok.com.ua